Учебник для вузов

ПИТЕР

СТАНДАРТ ТРЕТЬЕГО ПОКОЛЕНИЯ

В. Олифер Н. Олифер

Компьютерные сети

Принципы, технологии, протоколы

5-е издание

РЕКОМЕНДОВАНО
МИНИСТЕРСТВОМ ОБРАЗОВАНИЯ И НАУКИ РФ

СТАНДАРТ ТРЕТЬЕГО ПОКОЛЕНИЯ

В. Олифер, Н. Олифер

Компьютерные сети

Принципы, технологии, протоколы

5-е издание

Рекомендовано Министерством образования и науки Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению 552800 «Информатика и вычислительная техника» и по специальностям 220100 «Вычислительные машины, комплексы, системы и сети», 220200 «Автоматизированные системы обработки информации и управления» и 220400 «Программное обеспечение вычислительной техники и автоматизированных систем»

Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Киев · Екатеринбург · Самара · Минск

2016

ББК 32.973.202я7
УДК 004.7(075)
О-54

Рецензенты:

Кафедра «Вычислительная техника» факультета «Вычислительные машины и системы» Московского государственного института радиотехники, электроники и автоматики (Технического университета);

Ю. А. Григорьев, д. т. н., профессор кафедры «Системы обработки информации и управления» Московского государственного технического университета им. Н. Э. Баумана;

Б. Ф. Прижуков, к. т. н., заместитель начальника ИВЦ ОАО «Московский междугородный и международный телефон»

Олифер В., Олифер Н.

О-54 Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 5-е изд. — СПб.: Питер, 2016. — 992 с.: ил. — (Серия «Учебник для вузов»).

ISBN 978-5-496-01967-5

Пятое издание одного из лучших российских учебников по сетевым технологиям, переведенного на английский, испанский, португальский и китайский языки, отражает те изменения, которые произошли в области компьютерных сетей за 6 лет, прошедших со времени подготовки предыдущего издания: преодоление локальными и глобальными сетями рубежа скорости в 100 Гбит/с и освоение терабитных скоростей; повышение эффективности и гибкости первичных оптических сетей за счет появления реконфигурируемых мультиплексоров ввода-вывода (ROADM) и применения суперканалов DWDM, работающих на основе гибкого частотного плана; развитие техники виртуализации сетевых функций и услуг, приведшей к распространению облачных сервисов; выход на первый план проблем безопасности.

Издание предназначено для студентов, аспирантов и технических специалистов, которые хотели бы получить базовые знания о принципах построения компьютерных сетей, понять особенности традиционных и перспективных технологий локальных и глобальных сетей, изучить способы создания крупных составных сетей и управления такими сетями.

Рекомендовано Министерством образования и науки Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению «Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем».

ББК 32.973.202я7
УДК 004.7(075)

ISBN 978-5-496-01967-5

Оглавление

Посвящаем нашей дочери
Анне

От авторов

Эта книга является результатом многолетнего опыта преподавания авторами курсов сетевой тематики в аудиториях государственных вузов и различных учебных центров, а также участия в научно-технических разработках, таких как проект Janet, связанный с созданием объединяющей сети кампусов университетов и исследовательских центров Великобритании, и панъевропейские проекты GEANT2 и GEANT3.

Основу книги составили материалы курсов «Проблемы построения корпоративных сетей», «Основы сетевых технологий», «Организация удаленного доступа», «Сети TCP/IP», «Стратегическое планирование сетей масштаба предприятия» и ряда других. Эти материалы прошли успешную проверку в бескомпромиссной и сложной аудитории, состоящей из слушателей с существенно разным уровнем подготовки и кругом профессиональных интересов. Среди них были студенты и аспиранты вузов, сетевые администраторы и интеграторы, начальники отделов автоматизации и преподаватели. Учитывая специфику аудитории, курсы лекций строились так, чтобы начинающий получил основу для дальнейшего изучения, а специалист систематизировал и актуализировал свои знания. В соответствии с такими же принципами написана и эта книга — она является фундаментальным курсом по компьютерным сетям, который сочетает широту охвата основных областей, проблем и технологий этой быстроразвивающейся области знаний с основательным рассмотрением деталей каждой технологии.

Для кого эта книга

Книга предназначена для студентов, аспирантов и технических специалистов, которые хотят получить базовые знания о принципах построения компьютерных сетей, понять особенности традиционных и перспективных технологий локальных и глобальных сетей, изучить способы создания крупных составных сетей и управления такими сетями.

Учебник будет полезен начинающим специалистам в области сетевых технологий, которые имеют только общие представления о работе сетей из опыта общения с персональными компьютерами и Интернетом, но хотели бы получить фундаментальные знания, позволяющие продолжить изучение сетей самостоятельно.

Сложившимся сетевым специалистам книга может помочь в знакомстве с теми технологиями, с которыми им не приходилось сталкиваться в практической работе, систематизировать имеющиеся знания, стать справочником, позволяющим найти описание конкретного протокола, формата кадра и т. п. Кроме того, книга дает необходимую теоретическую основу для подготовки к сертификационным экзаменам таким, например, как Cisco CCNA, CCNP, CCDP и CCIP.

Студенты высших учебных заведений, обучающиеся по направлению «220000. Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем»,

могут использовать книгу в качестве рекомендованного Министерством образования Российской Федерации учебного пособия.

Изменения в пятом издании

За время, прошедшее с момента выхода предыдущего издания этой книги, появились новые сетевые технологии: локальные и глобальные сети взяли рубеж скорости в 100 Гбит/с и уже готовятся к терабитным скоростям; первичные оптические сети существенно повысили свою эффективность за счет появления реконфигурируемых мультиплексоров ввода-вывода (ROADM) и применения суперканалов DWDM, работающих на основе гибкого частотного плана; дальнейшее развитие получила техника виртуализации сетевых функций и услуг, нашедшая практическое применение в организации облачных сервисов. Описания этих и некоторых других новых технологий добавлены в 5-е издание книги.

Произошла также переоценка относительной важности различных аспектов функционирования компьютерных сетей — в первую очередь это проявилось в выходе на первый план проблем их безопасности. Сегодня этими вопросами озабочены как разработчики сетевых операционных систем и сервисов, старающиеся свести к минимуму риски перехвата или подмены конфиденциальных данных, так и разработчики транспортных технологий, пытающиеся обеспечить устойчивое функционирование компьютерной сети в условиях разнообразных распределенных атак, а также предотвратить использование транспортных протоколов в качестве орудия атаки. В новом издании вопросам безопасности компьютерных сетей отводится существенно больше места — вместо одной главы, как было в 4-м издании, теперь безопасности посвящено пять глав, в которых рассматриваются:

- Основные понятия, концепции и принципы информационной безопасности, такие как уязвимость, угроза, атака, ущерб, конфиденциальность, целостность, доступность, модель оценки рисков, а также методы криптографии, в том числе симметричные и асимметричные методы шифрования, способы распределения ключей, хэш-функции.
- Технологии аутентификации, авторизации и управления доступом; подробно рассматриваются мандатный, дискреционный и ролевой способы доступа, а также соответствующие механизмы операционных систем.
- Технологии безопасности на основе фильтрации и мониторинга трафика, реализуемые маршрутизаторами, файерволами, прокси-серверами, анализаторами протоколов, системами аудита и обнаружения вторжений.
- Атаки на транспортную инфраструктуру сети, включая рассмотрение уязвимостей и методов защиты каждого из базовых протоколов стека TCP/IP: IP, ICMP, TCP, UDP, BGP, а также службы DNS.
- Уязвимости программного кода и сетевых служб, вредоносные программы — трояны, черви, вирусы и программные закладки.

В новом издании авторы уделили больше внимания сетевым функциям операционных систем, относящимся к прикладному уровню. Помимо упомянутых систем аутентификации, авторизации и управления доступом, обеспечивающих безопасность сети, книга включает описание сетевых служб, предоставляющих услуги конечным пользователям и администраторам сети, таких как веб-служба, электронная почта, IP-телефония, сетевая файловая система, служба управления сетью, облачные сервисы.

Серьезной реструктуризации и переработке подверглись некоторые «традиционные» части книги, что, как надеются авторы, улучшило изложение материала.

Для сохранения приемлемого объема книги авторы применили тот же прием, что и при подготовке предыдущего издания, — некоторые разделы вынесены на веб-сайт поддержки данной книги www.olifer.co.uk. Материалы веб-сайта дополнены новыми разделами: «Операционная система маршрутизаторов Cisco iOS», «Биометрическая аутентификация», «Централизованная справочная служба» и другими. Для ссылки на материалы, помещенные на сайт, используется значок **(S)** в соответствующих местах книги. На сайт также вынесена значительная часть вопросов и ответов к ним.

И наконец, были исправлены ошибки и опечатки в тексте и рисунках, замеченные читателями и самими авторами.

Мы с благодарностью примем ваши отзывы по адресу victor.olifer@jisc.ac.uk и natalia@olifer.co.uk.

Благодарности

Мы благодарим наших читателей за их многочисленные пожелания, вопросы и замечания.

Мы признательны также всем сотрудникам издательства «Питер», которые принимали участие в создании этой книги. Особая благодарность президенту издательства «Питер» Вадиму Усманову и нашему неизменному литературному редактору Алексею Жданову.

Виктор Олифер, к. т. н., ССІР

Наталья Олифер, к. т. н., доцент

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу электронной почты comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства http://www.piter.com вы найдете подробную информацию о наших книгах.

Часть I

Основы сетей передачи данных

- Глава 1. Эволюция компьютерных сетей
- Глава 2. Общие принципы построения сетей
- Глава 3. Коммутация каналов и пакетов
- Глава 4. Архитектура, стандартизация и классификация сетей
- Глава 5. Сетевые характеристики
- Глава 6. Методы обеспечения качества обслуживания

Процесс познания всегда развивается по спирали. Мы не можем сразу понять и осознать сложное явление, мы должны рассматривать его с разных точек зрения, в целом и по частям, изолированно и во взаимодействии с другими явлениями, накапливая знания постепенно, время от времени возвращаясь к уже, казалось бы, понятому и с каждым новым витком все больше проникая в суть явления. Хорошим подходом является первоначальное изучение общих принципов некоторой области знаний с последующим детальным рассмотрением реализации этих принципов в конкретных методах, технологиях или конструкциях. Первая часть книги является таким «первым витком» изучения компьютерных сетей.

Изучение общих принципов построения компьютерных сетей поможет вам в дальнейшем быстрее «разбираться» с любой конкретной сетевой технологией. Однако известное высказывание «Знание нескольких принципов освобождает от запоминания множества фактов» не стоит воспринимать буквально — хороший специалист, конечно же, должен знать множество деталей и фактов. Знание принципов позволяет систематизировать эти частные сведения, связать их друг с другом в стройную систему и тем самым использовать более осознанно и эффективно. Конечно, изучение принципов перед изучением конкретных технологий — задача непростая, особенно для читателей с практическим складом ума. Кроме того, всегда есть опасность неверного понимания какого-нибудь общего утверждения без проверки его в практической реализации. Поэтому мы просим читателей поверить нам пока на слово, что игра стоит свеч, а также последовать нашему совету: в ходе изучения материала последующих глав книги время от времени мысленно возвращайтесь к теоретическим вопросам и проверяйте себя, так ли вы понимали те или иные механизмы, когда изучали их впервые.

Часть, а вместе с ней и книга, открывается главой об эволюции компьютерных сетей. История любой отрасли науки и техники позволяет не только удовлетворить естественное любопытство, но и глубже понять сущность основных достижений в этой отрасли, осознать существующие тенденции и правильно оценить перспективность тех или иных направлений развития.

В следующих двух главах рассматриваются фундаментальные концепции компьютерных сетей — коммутация, маршрутизация, мультиплексирование, адресация. Изучаются методы продвижения

пакетов — дейтаграммная передача, передача с установлением логического соединения и техника виртуальных каналов.

Важной темой данной части книги является рассматриваемая в четвертой главе стандартизация архитектуры компьютерной сети, идеологической основой которой служит модель взаимодействия открытых систем (OSI).

Две последние главы этой части книги посвящены сетевым характеристикам и проблемам качества обслуживания. Новая роль компьютерных сетей как основы для создания следующего поколения публичных сетей, предоставляющих все виды информационных услуг и переносящих данные, а также аудио- и видеотрафик, привела к проникновению методов обеспечения качества обслуживания практически во все коммуникационные технологии. Таким образом, концепции качества обслуживания, которые достаточно долго рассматривались как вспомогательное направление сетевой отрасли, вошли в число базовых принципов построения компьютерных сетей.

ГЛАВА 1 Эволюция компьютерных сетей

Два корня компьютерных сетей

Вычислительная техника и телекоммуникации

Компьютерные сети, которым посвящена данная книга, отнюдь не являются единственным видом сетей, созданным человеческой цивилизацией. Даже водопроводы Древнего Рима можно рассматривать как один из наиболее древних примеров сетей, покрывающих большие территории и обслуживающих многочисленных клиентов. Другой, менее экзотический пример — электрические сети. В них легко можно найти аналоги компонентов любой территориальной компьютерной сети: источникам информационных ресурсов соответствуют электростанции, магистралям — высоковольтные линии электропередачи, сетям доступа — трансформаторные подстанции, клиентским терминалам — осветительные и бытовые электроприборы.

Компьютерные сети, называемые также **сетями передачи данных**, являются логическим результатом эволюции двух важнейших научно-технических отраслей современной цивилизации — вычислительной техники и телекоммуникационных технологий.

С одной стороны, компьютерные сети представляют собой группу компьютеров, согласованно решающих набор взаимосвязанных задач, обмениваясь данными в автоматическом режиме. С другой стороны, компьютерные сети могут рассматриваться как средство передачи информации на большие расстояния, для чего в них применяются методы кодирования и мультиплексирования данных, получившие развитие в различных телекоммуникационных системах (рис. 1.1).

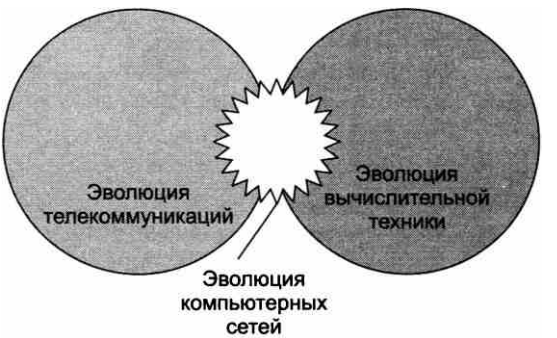


Рис. 1.1. Эволюция компьютерных сетей на стыке вычислительной техники и телекоммуникационных технологий

Системы пакетной обработки

Обратимся сначала к компьютерному корню вычислительных сетей. Первые компьютеры 50-х годов — большие, громоздкие и дорогие — предназначались для очень небольшого числа избранных пользователей. Часто эти монстры занимали целые здания. Такие компьютеры не были предназначены для интерактивной работы пользователя, а применялись в режиме пакетной обработки.

Системы пакетной обработки, как правило, строились на базе мэйнфрейма — мощного и надежного компьютера универсального назначения. Пользователи подготавливали перфокарты, содержащие данные и команды программ, и передавали их в вычислительный центр (рис. 1.2). Задания нескольких пользователей группировались в пакет, который принимался на выполнение. Оператор мэйнфрейма вводил карты пакета в компьютер, который обрабатывал задания в многопрограммном режиме, оптимизируя распределение процессора и устройств ввода-вывода между заданиями для достижения максимальной производительности вычислений. Распечатанные результаты пользователи получали обычно только на следующий день. Таким образом, одна неверно набитая карта означала как минимум суточную задержку. Конечно, для пользователей интерактивный режим работы, при котором можно с терминала оперативно руководить процессом обработки своих данных, был бы удобнее. Но интересами пользователей на первых этапах развития вычислительных систем в значительной степени пренебрегали. Во главу угла ставилась эффективность работы самого дорогого устройства вычислительной машины — процессора, даже в ущерб эффективности работы использующих его специалистов.

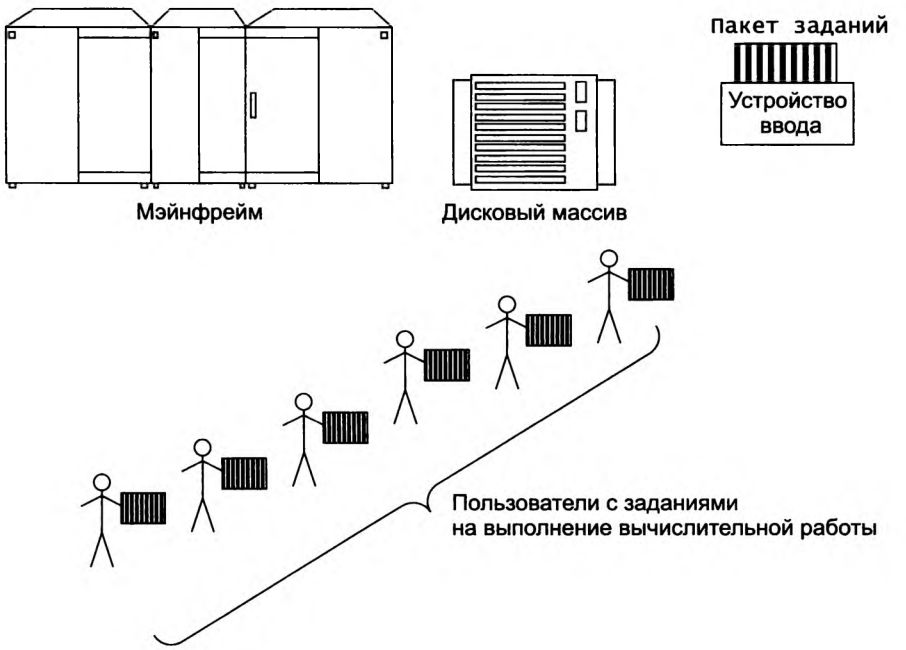


Рис. 1.2. Централизованная система на базе мэйнфрейма

Многотерминальные системы — прообраз сети

По мере удешевления процессоров в начале 60-х годов появились новые способы организации вычислительного процесса, которые позволили учесть интересы пользователей. Начали развиваться интерактивные **многотерминальные системы разделения времени** (рис. 1.3). В таких системах каждый пользователь получал собственный терминал, с помощью которого он мог вести диалог с компьютером. Количество одновременно работающих с компьютером пользователей определялось его мощностью: время реакции вычислительной системы должно было быть достаточно мало, чтобы пользователю была не слишком заметна параллельная работа с компьютером других пользователей.

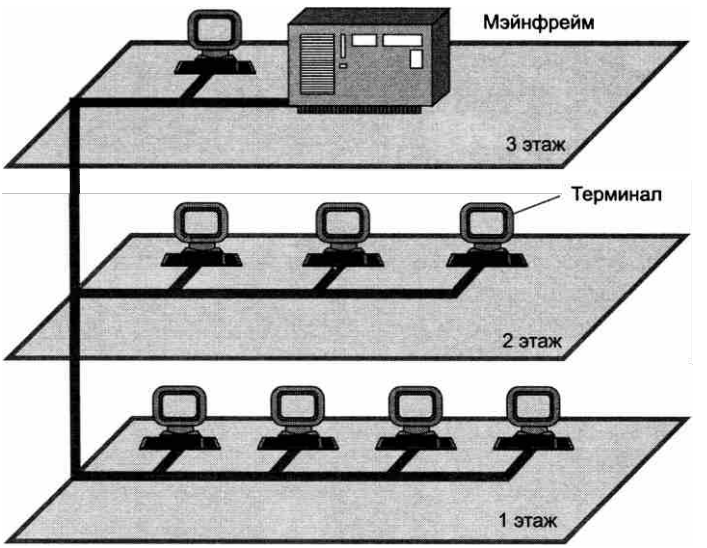


Рис. 1.3. Многотерминальная система — прообраз вычислительной сети

Терминалы, выйдя за пределы вычислительного центра, рассредоточились по всему предприятию. И хотя вычислительная мощность оставалась полностью централизованной, некоторые функции, такие как ввод и вывод данных, стали распределенными. Подобные многотерминальные централизованные системы внешне уже были очень похожи на локальные вычислительные сети. Действительно, рядовой пользователь работу за терминалом мэйнфрейма воспринимал примерно так же, как сейчас он воспринимает работу за подключенным к сети персональным компьютером. Пользователь мог получить доступ к общим файлам и периферийным устройствам, при этом у него поддерживалась полная иллюзия единоличного владения компьютером, так как он мог запустить нужную ему программу в любой момент и почти сразу же получить результат. (Некоторые далекие от вычислительной техники пользователи даже были уверены, что все вычисления выполняются внутри их дисплея.)

Многотерминальные системы, работающие в режиме разделения времени, стали прообразом локальных вычислительных сетей.

Однако до появления локальных сетей нужно было пройти еще большой путь, так как многотерминальные системы хотя и имели внешние черты распределенных систем, все еще поддерживали централизованную обработку данных.

К тому же потребность предприятий в создании локальных сетей в это время еще не созрела — в одном здании просто нечего было объединять в сеть, так как из-за высокой стоимости вычислительной техники предприятия не могли себе позволить роскошь приобретения нескольких компьютеров. В этот период был справедлив так называемый **закон Гроша**, который эмпирически отражал уровень технологии того времени. В соответствии с этим законом *производительность компьютера была пропорциональна квадрату его стоимости*. Отсюда следовало, что за одну и ту же сумму было выгоднее купить одну мощную машину, чем две менее мощных — их суммарная мощность оказывалась намного ниже мощности дорогой машины.

Первые компьютерные сети

Первые глобальные сети

А вот потребность в соединении нескольких компьютеров, находящихся на большом расстоянии друг от друга, к этому времени уже вполне назрела. Началось все с решения более простой задачи — доступа к отдельному компьютеру с терминалов, удаленных от него на многие сотни, а то и тысячи километров. Терминалы соединялись с компьютером через телефонные сети с помощью модемов, позволив многочисленным пользователям получать удаленный доступ к разделяемым ресурсам мощных суперкомпьютеров. Затем появились системы, в которых наряду с удаленными соединениями типа *терминал—компьютер* были реализованы и удаленные связи типа *компьютер—компьютер*.

Разнесенные территориально компьютеры получили возможность *обмениваться данными в автоматическом режиме*, что, собственно, и является базовым признаком любой вычислительной сети.

На основе подобного механизма в первых сетях были реализованы службы обмена файлами, синхронизации баз данных, электронной почты и другие ставшие теперь традиционными сетевые службы.

Итак, хронологически первыми появились **глобальные сети** (Wide Area Network, WAN), то есть сети, объединяющие территориально рассредоточенные компьютеры, возможно, находящиеся в различных городах и странах.

Именно при построении глобальных сетей были впервые предложены и отработаны многие основные идеи, лежащие в основе современных вычислительных сетей. Такие, например, как многоуровневое построение коммуникационных протоколов, концепции коммутации и маршрутизации пакетов.

Глобальные компьютерные сети очень многое унаследовали от других, гораздо более старых и распространенных глобальных сетей — *телефонных*. Главное технологическое новшество, которое привнесли с собой первые глобальные компьютерные сети, состояло в отказе от принципа коммутации каналов, на протяжении многих десятков лет успешно использовавшегося в телефонных сетях.

Выделяемый на все время сеанса связи составной телефонный канал, передающий информацию с постоянной скоростью, не мог эффективно использоваться пульсирующим трафиком компьютерных данных, у которого периоды интенсивного обмена чередуются с продолжительными паузами. Натурные эксперименты и математическое моделирование показали, что пульсирующий и в значительной степени не чувствительный к задержкам компьютерный трафик гораздо эффективнее передается сетями, работающими по принципу коммутации пакетов, когда данные разделяются на небольшие порции — пакеты, которые самостоятельно перемещаются по сети благодаря наличию адреса конечного узла в заголовке пакета.

Так как прокладка высококачественных линий связи на большие расстояния обходится очень дорого, то в первых глобальных сетях часто использовались уже существующие каналы связи, изначально предназначенные совсем для других целей. Например, в течение многих лет глобальные сети строились на основе телефонных каналов тональной частоты, способных в каждый момент времени вести передачу только одного разговора в аналоговой форме. Поскольку скорость передачи дискретных компьютерных данных по таким каналам была очень низкой (десятки килобит в секунду), набор предоставляемых услуг в глобальных сетях подобного типа обычно ограничивался передачей файлов (преимущественно в фоновом режиме) и электронной почтой. Помимо низкой скорости такие каналы имеют и другой недостаток — они вносят значительные искажения в передаваемые сигналы. Поэтому протоколы глобальных сетей, построенных с использованием каналов связи низкого качества, отличались сложными процедурами контроля и восстановления данных. Типичным примером таких сетей являются сети X.25, разработанные еще в начале 70-х годов.

ПРИМЕЧАНИЕ

При написании этой главы авторы столкнулись с дилеммой: невозможно рассказывать об истории отрасли, не называя конкретные технологии и концепции. Но в то же время невозможно давать пояснения этих технологий и концепций, так как читатель, перелистывающий первые страницы, еще не готов к восприятию объяснений. Авторы пошли по пути компромисса, отложив на будущее исчерпывающие пояснения многих терминов ради того, чтобы в самом начале изучения компьютерных сетей читатель имел возможность представить картину эволюции компьютерных сетей во всем ее красочном многообразии. И конечно, было бы очень полезно вернуться к этой главе после того, как будет перевернута последняя страница книги, чтобы, вооружившись новыми знаниями, сделать качественно новую попытку оценить прошлое и будущее компьютерных сетей.

В 1969 году министерство обороны США инициировало работы по объединению в единую сеть суперкомпьютеров оборонных и научно-исследовательских центров. Эта сеть, получившая название ARPANET, стала отправной точкой для создания первой и самой известной ныне глобальной сети мирового масштаба — Internet.

Сеть ARPANET объединяла компьютеры разных типов, работавшие под управлением различных операционных систем (ОС) с дополнительными модулями, реализующими коммуникационные протоколы, общие для всех компьютеров сети. ОС этих компьютеров можно считать *первыми* **сетевыми операционными системами**.

Сетевые ОС позволили не только рассредоточить пользователей между несколькими компьютерами (как в многотерминальных системах), но и организовать распределенные хранение и обработку данных. Любая сетевая операционная система, с одной стороны,

выполняет все функции локальной операционной системы, а с другой — обладает некоторыми дополнительными средствами, позволяющими ей взаимодействовать через сеть с операционными системами других компьютеров. Программные модули, реализующие сетевые функции, появлялись в операционных системах постепенно, по мере развития сетевых технологий, аппаратной базы компьютеров и возникновения новых задач, требующих сетевой обработки.

Прогресс глобальных компьютерных сетей во многом определялся прогрессом телефонных сетей.

С конца 60-х годов в телефонных сетях все чаще стала применяться передача голоса в цифровой форме.

Это привело к появлению высокоскоростных цифровых каналов, соединяющих автоматические телефонные станции (АТС) и позволяющих одновременно передавать десятки и сотни разговоров.

К настоящему времени глобальные сети по разнообразию и качеству предоставляемых услуг догнали локальные сети, которые долгое время лидировали в этом отношении, хотя и появились на свет значительно позже.

Первые локальные сети

Важное событие, повлиявшее на эволюцию компьютерных сетей, произошло в начале 70-х годов. В результате технологического прорыва в области производства компьютерных компонентов появились **большие интегральные схемы** (БИС). Их сравнительно невысокая стоимость и хорошие функциональные возможности привели к созданию **мини-компьютеров**, которые стали реальными конкурентами мэйнфреймов. Эмпирический закон Гроша перестал соответствовать действительности, так как десяток мини-компьютеров, имея ту же стоимость, что и один мэйнфрейм, решали некоторые задачи (как правило, хорошо распараллеливаемые) быстрее.

Даже небольшие подразделения предприятий получили возможность иметь собственные компьютеры. Мини-компьютеры решали задачи управления технологическим оборудованием, складом и другие задачи уровня отдела предприятия. Таким образом, появилась концепция распределения компьютерных ресурсов по всему предприятию. Однако при этом все компьютеры одной организации по-прежнему продолжали работать *автономно* (рис. 1.4).

Шло время, и потребности пользователей вычислительной техники росли. Их уже не удовлетворяла изолированная работа на собственном компьютере, им хотелось в автоматическом режиме обмениваться компьютерными данными с пользователями других подразделений. Ответом на эту потребность стало появление первых локальных вычислительных сетей (рис. 1.5).

Локальные сети (Local Area Network, LAN) — это объединения компьютеров, сосредоточенных на небольшой территории, обычно в радиусе не более 1–2 км, хотя в отдельных случаях локальная сеть может иметь и большие размеры, например несколько десятков километров. В общем случае локальная сеть представляет собой коммуникационную систему, принадлежащую одной организации.

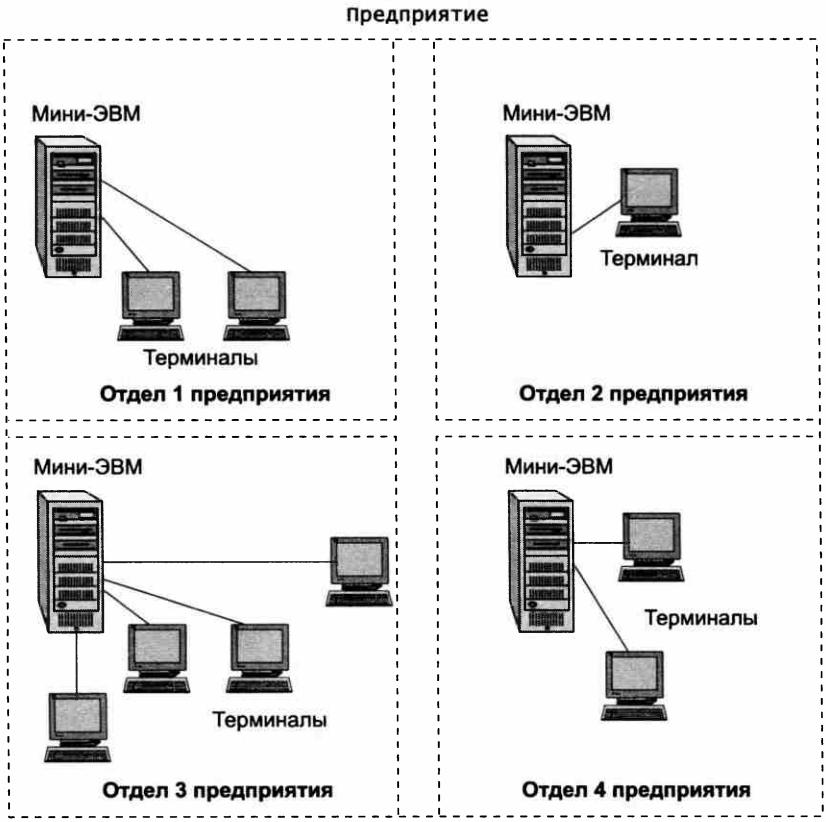


Рис. 1.4. Автономное использование нескольких мини-компьютеров на одном предприятии

На первых порах для соединения компьютеров друг с другом использовались нестандартные сетевые технологии.

Сетевая технология — это согласованный набор программных и аппаратных средств (например, драйверов, сетевых адаптеров, кабелей и разъемов), а также механизмов передачи данных по линиям связи, достаточный для построения вычислительной сети.

Разнообразные устройства сопряжения, использующие собственные способы представления данных на линиях связи, свои типы кабелей и т. п., могли соединять только те конкретные модели компьютеров, для которых были разработаны, например, мини-компьютеры PDP-11 с мэйнфреймом IBM 360 или мини-компьютеры HP с микрокомпьютерами LSI-11. Такая ситуация создала большой простор для творчества студентов — названия многих курсовых и дипломных проектов начинались тогда со слов «Устройство сопряжения...».

В середине 80-х годов положение дел в локальных сетях кардинально изменилось. Утвердились **стандартные сетевые технологии** объединения компьютеров в сеть — Ethernet, Arcnet, Token Ring, Token Bus, несколько позже — FDDI.

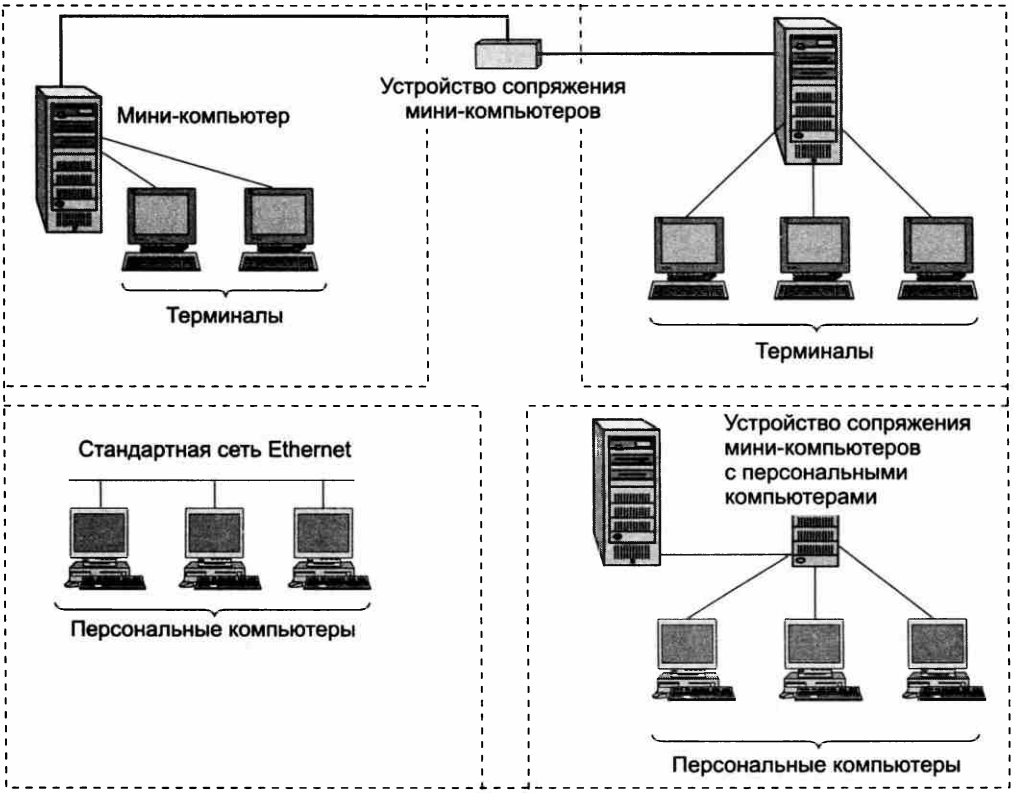


Рис. 1.5. Различные типы связей в первых локальных сетях

Мощным стимулом для их появления послужили **персональные компьютеры**. Эти массовые продукты стали идеальными элементами построения сетей — с одной стороны, они были достаточно мощными, чтобы обеспечивать работу сетевого программного обеспечения, а с другой — явно нуждались в объединении своей вычислительной мощности для решения сложных задач, а также разделения дорогих периферийных устройств и дисковых массивов. Поэтому персональные компьютеры стали преобладать в локальных сетях, причем не только в качестве клиентских компьютеров, но и в качестве центров хранения и обработки данных, то есть сетевых серверов, потеснив с этих привычных ролей мини-компьютеры и мэйнфреймы.

Все стандартные технологии локальных сетей опирались на тот же принцип коммутации, который был с успехом опробован и доказал свои преимущества при передаче трафика данных в глобальных компьютерных сетях, — принцип коммутации пакетов.

Стандартные сетевые технологии превратили процесс построения локальной сети из решения нетривиальной технической проблемы в рутинную работу. Для создания сети достаточно было приобрести стандартный кабель, сетевые адаптеры соответствующего стандарта, например Ethernet, вставить адаптеры в компьютеры, присоединить их к кабелю стандартными разъемами и установить на компьютеры одну из популярных сетевых операционных систем, например Novell NetWare.

Разработчики локальных сетей привнесли много нового в организацию работы пользователей. Так, стало намного проще и удобнее, чем в глобальных сетях, получать доступ к общим сетевым ресурсам. Последствием и одновременно движущей силой такого прогресса стало появление огромного числа непрофессиональных пользователей, освобожденных от необходимости изучать специальные (и достаточно сложные) команды для сетевой работы.

Конец 90-х выявил явного лидера среди технологий локальных сетей — семейство Ethernet, в которое вошли классическая технология Ethernet со скоростью передачи 10 Мбит/с, а также Fast Ethernet со скоростью 100 Мбит/с и Gigabit Ethernet со скоростью 1000 Мбит/с.

Простые алгоритмы работы этой технологии предопределяют низкую стоимость оборудования Ethernet. Широкий диапазон иерархии скоростей позволяет рационально строить локальную сеть, выбирая ту технологию семейства, которая в наибольшей степени отвечает задачам предприятия и потребностям пользователей. Важно также, что все технологии Ethernet очень близки друг к другу по принципам работы, что упрощает обслуживание и интеграцию этих сетей.

Конвергенция сетей

Сближение локальных и глобальных сетей

В конце 80-х годов отличия между локальными и глобальными сетями проявлялись весьма отчетливо.

- *Протяженность и качество линий связи.* Локальные компьютерные сети по определению отличаются от глобальных сетей небольшими расстояниями между узлами сети. Это в принципе делает возможным использование в локальных сетях более качественных линий связи. В глобальных сетях 80-х годов преобладали низкоскоростные телефонные линии связи, передающие дискретную информацию компьютеров со сравнительно частыми искажениями.
- *Сложность методов передачи данных.* В условиях низкой надежности физических каналов в глобальных сетях требуются более сложные, чем в локальных сетях, методы передачи данных и соответствующее оборудование.
- *Скорость обмена данными* в локальных сетях (10, 16 и 100 Мбит/с) в то время была существенно выше, чем в глобальных (от 2,4 Кбит/с до 2 Мбит/с).
- *Разнообразие услуг.* Высокие скорости обмена данными позволили предоставлять в локальных сетях широкий спектр услуг — это прежде всего разнообразные механизмы использования файлов, хранящихся на дисках других компьютеров сети, совместное использование устройств печати, модемов, факсов, доступ к единой базе данных, электронная почта и другие. В то же время глобальные сети в основном ограничивались почтовыми и файловыми услугами в их простейшем (не самом удобном для пользователя) виде.

Постепенно различия между локальными и глобальными сетевыми технологиями стали сглаживаться. Изолированные ранее локальные сети начали объединять друг с другом, при этом в качестве связующей среды использовались глобальные сети. Тесная интеграция

локальных и глобальных сетей привела к значительному взаимопроникновению соответствующих технологий.

Сближение в методах передачи данных происходит на платформе цифровой передачи данных по волоконно-оптическим линиям связи. Эта среда передачи используется практически во всех технологиях локальных сетей для скоростного обмена информацией на расстояниях свыше 100 метров, на ней же стали строиться магистрали первичных сетей SDH и DWDM, предоставляющих свои цифровые каналы для объединения оборудования глобальных компьютерных сетей.

Высокое качество цифровых каналов изменило требования к протоколам глобальных компьютерных сетей. На первый план вместо процедур обеспечения надежности вышли процедуры обеспечения гарантированной средней скорости доставки информации пользователям, а также механизмы приоритетной обработки пакетов особенно чувствительного к задержкам трафика, например голосового. Эти изменения нашли отражение в таких технологиях глобальных сетей 90-х годов, как Frame Relay и ATM. В этих технологиях предполагается, что искажение битов происходит настолько редко, что ошибочный пакет выгоднее просто уничтожить, а все проблемы, связанные с его потерей, перепоручить программному обеспечению более высокого уровня, которое непосредственно не входит в состав сетей Frame Relay и ATM.

Большой вклад в сближение локальных и глобальных сетей внесло доминирование протокола IP. Этот протокол может работать поверх любых технологий локальных и глобальных сетей (Ethernet, MPLS, Token Ring, ATM, Frame Relay), объединяя различные подсети в единую составную сеть.

Начиная с 90-х годов компьютерные глобальные сети, работающие на основе скоростных цифровых каналов, существенно расширили спектр предоставляемых услуг и догнали в этом отношении локальные сети. Стало возможным создание служб, работа которых связана с доставкой пользователю больших объемов информации в реальном времени — изображений, видеофильмов, голоса, в общем, всего того, что получило название мультимедийной информации. Наиболее яркий пример — гипертекстовая информационная служба World Wide Web (веб-служба), ставшая основным поставщиком информации в Интернете. Ее интерактивные возможности превзошли возможности многих аналогичных служб локальных сетей, так что разработчикам приложений локальных сетей пришлось просто позаимствовать эту службу у глобальных сетей. Процесс переноса технологий из глобальной сети Интернет в локальные приобрел такой массовый характер, что появился даже специальный термин — **intranet-технологии** (intra — внутренний).

Возникли новые транспортные технологии, которые стали одинаково успешно работать как в локальных, так и в глобальных сетях. Первой такой технологией была ATM, которая могла эффективно объединять все существующие типы трафика в одной транспортной сети. Однако истинно универсальной транспортной технологией стала технология Ethernet. Долгие годы Ethernet был технологией только локальных сетей, однако дополненная новыми функциями и новыми уровнями скоростей, эта технология (называемая в этом варианте Carrier Ethernet, то есть Ethernet операторского класса) сегодня преобладает на линиях связи и глобальных сетей. Следствием доминирования технологии Ethernet в первом десятилетии XXI века стало упрощение структуры как локальных, так и глобальных сетей — в подавляющем большинстве подсетей сегодня работает протокол Ethernet, а объединяются подсети в составную сеть с помощью протокола IP.

Еще одним признаком сближения локальных и глобальных сетей является появление сетей, занимающих промежуточное положение между локальными и глобальными сетями. **Городские сети**, или **сети мегаполисов** (Metropolitan Area Network, MAN), предназначены для обслуживания территории крупного города.

Эти сети используют цифровые линии связи, часто оптоволоконные, со скоростями на магистрали 10 Гбит/с и выше. Они обеспечивают экономичное соединение локальных сетей между собой, а также выход в глобальные сети. Сети MAN первоначально были разработаны только для передачи данных, но сейчас перечень предоставляемых ими услуг расширился, в частности они поддерживают видеоконференции и интегральную передачу голоса и текста. Современные сети MAN отличаются разнообразием предоставляемых услуг, позволяя своим клиентам объединять коммуникационное оборудование различного типа, в том числе офисные АТС.

Новой вехой на пути конвергенции сетей обещают стать так называемые **облачные вычисления**, которые позволяют разгрузить пользовательский компьютер и перенести выполнение приложений на некоторые удаленные компьютеры, связанные с пользовательским компьютером через сеть.

Конвергенция компьютерных и телекоммуникационных сетей

Начиная с 1980-х годов предпринимаются попытки создания универсальной, так называемой **мультисервисной сети**, способной предоставлять услуги как компьютерных, так и телекоммуникационных сетей.

К телекоммуникационным сетям относятся радиосети, телефонные и телевизионные сети. Главное, что объединяет их с компьютерными сетями, — это то, что в качестве ресурса, предоставляемого клиентам, выступает информация. Однако имеется некоторая специфика, касающаяся вида, в котором представляют информацию компьютерные и телекоммуникационные сети. Так, изначально компьютерные сети разрабатывались для передачи алфавитно-цифровой информации, которую часто называют просто *данными*, поэтому у компьютерных сетей имеется и другое название — **сети передачи данных**, в то время как телекоммуникационные сети были созданы для передачи только *голосовой информации* (и изображения в случае телевизионных сетей).

Сегодня мы являемся свидетелями конвергенции телекоммуникационных и компьютерных сетей, которая идет по нескольким направлениям.

Прежде всего наблюдается *сближение видов услуг*, предоставляемых клиентам. Первая попытка создания мультисервисной сети, способной оказывать различные услуги, в том числе услуги телефонии и передачи данных, привела к появлению в 80-х годах технологии **цифровых сетей с интегрированным обслуживанием** (Integrated Services Digital Network, ISDN).

Однако на практике ISDN предоставляет сегодня в основном телефонные услуги, а на роль глобальной **мультисервисной сети нового поколения**, часто называемой в англоязычной литературе Next Generation Network (NGN), или New Public Network (NPN), претендует **Интернет**.

Интернет уже сегодня превратился из сети, предназначенной для оказания небольшого набора услуг передачи данных, основными из которых были передача файлов и обмен текстовыми почтовыми сообщениями, в действительно мультисервисную сеть. Интернет может оказывать все виды телекоммуникационных услуг, в том числе услуг мгновенных сообщений, видео- и аудиоконференций, IP-телефонии, IP-телевидения, а также услуг многочисленных социальных сетей. Очевидно, что мультисервисность Интернета в будущем будет только возрастать.

Прорывом в процессе конвергенции сетей явилось появление **смартфонов** — терминальных устройств, которые объединили в себе функции мобильных телефонов и персональных компьютеров. Для поддержки таких новых функций телефона современная мобильная телефонная сеть также стала истинной мультисервисной сетью — она предоставляет полный набор как телефонных, так и компьютеризованных информационных услуг (просмотр веб-страниц в такой же удобной форме, как и на экране компьютера, услуги электронной почты и видеоконференций, просмотр фильмов, публикация информации в социальных сетях и т. п.).

Технологическое сближение сетей происходит сегодня на основе цифровой передачи информации различного типа, метода коммутации пакетов и программирования услуг. Важным шагом телефонии навстречу компьютерным сетям было прежде всего представление голоса в цифровой форме, что сделало принципиально возможным передачу телефонного и компьютерного трафиков по одним и тем же цифровым каналам (телевидение также может сегодня передавать изображение в цифровой форме). Телефонные сети широко используют комбинацию методов коммутации каналов и пакетов. Так, для передачи служебных сообщений (называемых сообщениями сигнализации) применяются протоколы коммутации пакетов, аналогичные протоколам компьютерных сетей, а для передачи собственно голоса между абонентами коммутируется традиционный составной канал.

Сегодня пакетные методы коммутации постепенно теснят традиционные для телефонных сетей методы коммутации каналов даже при передаче голоса. У этой тенденции есть достаточно очевидная причина — на основе метода коммутации пакетов можно более эффективно использовать пропускную способность каналов связи и коммутационного оборудования. Например, паузы в телефонном разговоре могут составлять до 40 % общего времени соединения, однако только пакетная коммутация позволяет «вырезать» паузы и использовать высвободившуюся пропускную способность канала для передачи трафика других абонентов. Другой веской причиной перехода к коммутации пакетов является популярность Интернета — сети, построенной на основе данной технологии.

Обращение к технологии коммутации пакетов для одновременной передачи через пакетные сети разнородного трафика — голоса, видео и текста — сделало актуальной разработку новых методов обеспечения требуемого **качества обслуживания** (Quality of Service, QoS). Методы QoS призваны минимизировать уровень задержек для чувствительного к ним трафика, например голосового, и одновременно гарантировать среднюю скорость и динамичную передачу пульсаций для трафика данных.

Однако неверно было бы говорить, что методы коммутации каналов морально устарели и у них нет будущего. На новом витке спирали развития они находят свое применение, но уже в новых технологиях, таких как технологии первичных сетей, служащих основой как для компьютерных, так и телефонных сетей: Optical Transport Networks (OTN) и Dense Wavelength Division Multiplexing (DWDM).

Компьютерные сети многое позаимствовали у телефонных и телевизионных сетей. В частности, они взяли на вооружение методы обеспечения отказоустойчивости телефонных сетей, за счет которых последние демонстрируют высокую степень надежности, так недостающую порой Интернету и корпоративным сетям.

Телефонные сети, в свою очередь, многое перенимают у компьютерных сетей. Особенно это заметно на примере мобильных телефонных сетей, которые стали широко использовать протокол IP в своих технологиях 3-го и 4-го поколений (3G и 4G).

Сегодня становится все более очевидным, что мультисервисная сеть нового поколения не может быть создана в результате «победы» какой-нибудь одной технологии или одного подхода. Ее может породить только процесс конвергенции, когда от каждой технологии берется все самое лучшее и соединяется в некоторый новый сплав, который и обеспечивает требуемое качество для поддержки существующих и создания новых услуг. Появившийся термин — **инфокоммуникационная сеть** — прямо говорит о двух составляющих современной сети — информационной (компьютерной) и телекоммуникационной. Учитывая, что этот термин еще не приобрел достаточной популярности, мы будем использовать устоявшийся термин «телекоммуникационная сеть» в расширенном понимании — то есть включать в него и компьютерные сети.

Интернет как фактор развития сетевых технологий

Интернет является самой быстрорастущей технической системой в истории человечества. Интернет растет постоянно начиная с 80-х годов и в соответствии с прогнозами специалистов будет продолжать расти. «Размеры» Интернета можно оценивать по-разному, чаще всего используют такие показатели, как количество подключенных к Интернету терминальных устройств (компьютеров различных типов, планшетов, мобильных телефонов), количество пользователей, объем трафика, передаваемый в единицу времени.

На рис. 1.6 показан график роста числа пользователей Интернета за 40 лет существования этой сети. К 2014 году их число превысило 3 миллиарда, что составляет 42 % населения земного шара.

Количество терминальных устройств, выполняющих функции серверов (без учета пользовательских устройств), росло примерно такими же темпами: в 1980 году насчитывалось около 1000 хостов, подключенных к Интернету, в 1991 — более 1 000 000, в начале 2000-х — около 100 000 000 и, наконец, в 2013 — свыше 1 миллиарда. С учетом пользовательских устройств (настольных компьютеров, ноутбуков, планшетов и мобильных телефонов) общее количество терминальных устройств, подключенных к Интернету, составило в 2013 году 12 миллиардов.

Абсолютно взрывным оказался рост объема трафика (количество байтов, переданных в месяц через магистрали Интернета):

- 1990 — 1 TB (1 терабайт = 10^{12} байт, или 1000 гигабайт);
- 1996 — 2000 TB;
- 2000 — 84 PB (1 петабайт = 1000 терабайт);
- 2008 — 10 EB (1 экзабайт = 1000 петабайт);
- 2013 — 50 EB.

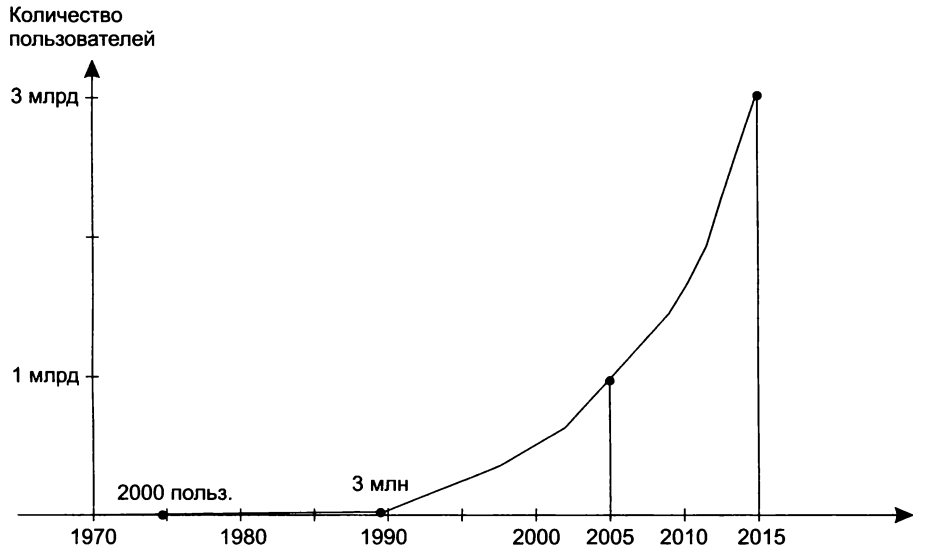


Рис. 1.6. Рост числа пользователей Интернета

В середине 90-х трафик рос особенно быстро, удваиваясь каждый год, то есть демонстрируя экспоненциальный рост. Затем рост несколько замедлился, но все равно за последние 5 лет объем передаваемого трафика вырос в 5 раз. Трафик рос не только в количественном отношении — существенно менялся процентный состав приложений, генерирующих трафик. Так, если в 90-е годы и начале 2000-х в общем объеме преобладал трафик приложений, передающих файлы (файлы электронной почты, веб-страниц, музыки и кинофильмов), то уже к 2010 году он уступил лидерство трафику приложений, передающих видеопотоки в реальном масштабе времени (таких, как интернет-телевидение, просмотр кинофильмов в онлайновом режиме по требованию, видеоконференции). *Изменение характера трафика* породило новые вызовы разработчикам сетевых технологий, так как требования к характеристикам сети у этих приложений значительно отличаются от требований приложений передачи файлов.

Еще одним революционным изменением в области передаваемого трафика стало резкое увеличение его доли, генерируемой *мобильными устройствами* — планшетами и мобильными телефонами. И если пока еще большая часть трафика генерируется персональными компьютерами (67 % в 2013 году), то к 2018 году эта доля, по прогнозам, упадет до 43 %, остальное будут генерировать мобильные устройства, а также компьютеры, прямо обменивающиеся данными между собой.

Такой феноменальный рост и изменчивость Интернета (в различных аспектах) оказывали и оказывают сильнейше влияние на технологии компьютерных сетей, заставляя их постоянно изменяться и совершенствоваться, приспосабливаясь к новым требованиям пользователей и их количеству. Эту движущую силу нужно учитывать при изучении любых технологий компьютерных сетей, основные из которых рассматриваются в последующих главах этой книги. А пока для иллюстрации того, как технологии отвечали на вызовы роста, ограничимся таким понятным показателем, как скорость передачи данных транспортными сетевыми технологиями, и посмотрим, как она изменялась в локальных и глобальных сетях:

- 80-е годы:
  - Локальные сети: большинство сетей используют Ethernet 10 Мбит/с, Token Ring 16 Мбит/с.
  - Глобальные сети: магистраль Интернета построена на цифровых телефонных каналах 56 Кбит/с; магистрали телефонных сетей используют цифровые линии 35–45 Мбит/с.
- 90-е годы:
  - Локальные сети: переход на 100 Мбит/с (FDDI и Fast Ethernet).
  - Глобальные сети: магистрали SDH 155 и 622 Мбит/с начинают применяться в Интернете.
- конец 90-х — начало 2000-х:
  - Локальные сети: в 1998 году появляется Gigabit Ethernet (1000 Мбит/с) и уже через четыре года, в 2002 году, — 10G Ethernet (10 Гбит/с).
  - Глобальные сети: иерархия скоростей SDH повышается до 10 Гбит/с; технология DWDM позволяет мультиплексировать в одном оптическом волокне до 40–80 каналов по 10 Гбит/с (общая пропускная способность волокна составляет 400–800 Гбит/с).
- Начало 2010-х:

  Локальные и глобальные сети: 40G и 100G Ethernet стандартизованы в 2012 году, версия 40G начинает применяться в серверах, а 100G — на магистралях сетей.

Как видно из этой краткой хронологии, разработчики сетевых транспортных технологий хорошо справлялись со своими обязанностями и смогли за 35 лет повысить потолок скорости в 10 000 раз.

Подводя итог, перечислим последовательность важнейших событий, ставших историческими вехами на пути эволюции компьютерных сетей (табл. 1.1).

Таблица 1.1. Хронология важнейших событий на пути появления первых компьютерных сетей

Этап	Время
Первые глобальные связи компьютеров, первые эксперименты с пакетными сетями	Конец 60-х
Начало передач по телефонным сетям голоса в цифровой форме	Конец 60-х
Появление больших интегральных схем, первые мини-компьютеры, первые нестандартные локальные сети	Начало 70-х
Стандартизация технологии X.25 для построения сети «удаленные терминалы — мейнфрейм»	1974
Появление персональных компьютеров, создание Интернета в современном виде, установка на всех узлах стека TCP/IP	Начало 80-х
Появление стандартных технологий локальных сетей (Ethernet — 1980 г., Token Ring, FDDI — 1985 г.)	Середина 80-х
Начало коммерческого использования Интернета	Конец 80-х

Этап	Время
Появление первичных сетей SONET/SDH со скоростью передачи до 155 Мбит/с	Конец 80-х
Изобретение Web	1991
Доминирование Ethernet в локальных сетях, стандартизация Gigabit Ethernet	Конец 90-х
Появление технологии плотного мультиплексирования волн (DWDM) с возможностью передачи 40/80 волн в одном волокне	Конец 90-х
Появление первых смартфонов с ограниченными интернет-функциями	Конец 90-х
Интернет становится мультимедийным (IP TV, IP-телефония)	Конец 90-х — начало 2000-х
Повышение скорости передачи данных до 10 Гбит/с (10G Ethernet и 10G SDH/OTN)	Начало 2000-х
Смартфоны становятся полнофункциональными интернет-терминалами	Середина 2000-х
Повышение скорости передачи до 100 Гбит/с (100G Ethernet и 100G OTN)	Начало 2010-х

Выводы

Компьютерные сети стали логическим результатом эволюции вычислительной техники и телекоммуникационных технологий.

Прообразом локальных вычислительных сетей являются многотерминальные системы, работающие в режиме разделения времени.

Хронологически первыми появились глобальные сети (Wide Area Network, WAN), то есть сети, объединяющие территориально рассредоточенные компьютеры, возможно, находящиеся в различных городах и странах.

Для связывания компьютеров в сеть операционные системы, установленные на них, были дополнены модулями, которые реализовали коммуникационные протоколы, общие для всех компьютеров сети. Такие ОС можно считать первыми сетевыми операционными системами. Сетевые ОС позволили не только рассредоточить пользователей между несколькими компьютерами (как в многотерминальных системах), но и организовать распределенные хранение и обработку данных.

В начале 70-х годов начались работы по созданию первой и самой известной ныне глобальной сети мирового масштаба — Internet.

Важнейший этап в развитии сетей — появление стандартных сетевых технологий: Ethernet, FDDI, Token Ring, позволяющих быстро и эффективно объединять компьютеры различных типов.

Начиная с 80-х годов стала проявляться тенденция сближения технологий локальных и глобальных компьютерных сетей, а также технологий телекоммуникационных сетей разных типов: телефонных, радио, телевизионных. В настоящее время ведутся активные работы по созданию универсальных мультисервисных сетей, способных одинаково эффективно передавать информацию любого типа: данные, голос и видео.

Феноменальный рост количества узлов и трафика Интернета, появление мобильных терминальных устройств — планшетов и смартфонов — оказывали и оказывают сильнейшее влияние на технологии компьютерных сетей, заставляя их постоянно изменяться и совершенствоваться, приспосабливаясь к новым требованиям пользователей.

Контрольные вопросы

1. Что было унаследовано компьютерными сетями от вычислительной техники, а что от телефонных сетей?
2. Какие свойства многотерминальной системы отличают ее от компьютерной сети?
3. В чем технология коммутации пакетов превосходит технологию коммутации каналов?
4. Каким образом развитие Интернета влияет на развитие сетевых технологий?
5. Поясните, почему глобальные компьютерные сети появились раньше локальных.

ГЛАВА 2 Общие принципы построения сетей

Простейшая сеть из двух компьютеров

Совместное использование ресурсов

Исторически главной целью объединения компьютеров в сеть было *разделение ресурсов*: пользователи компьютеров, подключенных к сети, или приложения, выполняемые на этих компьютерах, получают возможность автоматического доступа к разнообразным ресурсам остальных компьютеров сети, к числу которых относятся:

- периферийные устройства, такие как диски, принтеры, плоттеры, сканеры и др.;
- данные, хранящиеся в оперативной памяти или на внешних запоминающих устройствах;
- вычислительная мощность (за счет удаленного запуска «своих» программ на «чужих» компьютерах).

Чтобы обеспечить пользователей разных компьютеров возможностью совместного использования ресурсов сети, компьютеры необходимо оснастить некими дополнительными *сетевыми* средствами.

Рассмотрим простейшую сеть, состоящую из двух компьютеров, к одному из которых подключен принтер (рис. 2.1). Какие дополнительные средства должны быть предусмотрены в обоих компьютерах, чтобы с принтером мог работать не только пользователь компьютера *B*, к которому этот принтер непосредственно подключен, но и пользователь компьютера *A*?

Рис. 2.1. Простейшая сеть

Сетевые интерфейсы

Для связи устройств в них прежде всего должны быть предусмотрены внешние[1] интерфейсы.

---

[1] Наряду с внешними электронные устройства могут использовать внутренние интерфейсы, определяющие логические и физические границы между входящими в их состав модулями. Так, известный интерфейс «общая шина» является внутренним интерфейсом компьютера, связывающим оперативную память, процессор и другие блоки компьютера.

Интерфейс — в широком смысле — формально определенная логическая и/или физическая граница между взаимодействующими независимыми объектами. Интерфейс задает параметры, процедуры и характеристики взаимодействия объектов.

Разделяют физический и логический интерфейсы.

- **Физический интерфейс** (называемый также **портом**) определяется набором электрических связей и характеристиками сигналов. Обычно он представляет собой разъем с набором контактов, каждый из которых имеет определенное назначение, например это может быть группа контактов для передачи данных, контакт синхронизации данных и т. п. Пара разъемов соединяется **кабелем**, состоящим из набора проводов, каждый из которых соединяет соответствующие контакты. В таких случаях говорят о создании **линии**, или **канала**, **связи** между двумя устройствами.
- **Логический интерфейс** (называемый также **протоколом**) — это набор информационных сообщений определенного формата, которыми обмениваются два устройства или две программы, а также набор правил, определяющих логику обмена этими сообщениями.

На рис. 2.2 мы видим интерфейсы двух типов: компьютер—компьютер и компьютер—периферийное устройство.

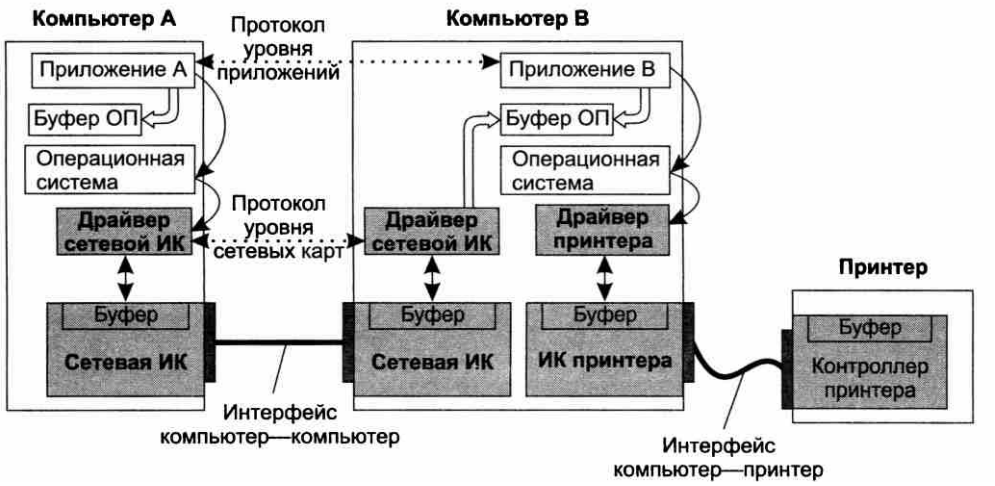


Рис. 2.2. Совместное использование принтера в компьютерной сети

- *Интерфейс компьютер—компьютер* позволяет двум компьютерам обмениваться информацией. С каждой стороны он реализуется парой:
  - аппаратным модулем, называемым **сетевым адаптером**, или **сетевой интерфейсной картой** (Network Interface Card, NIC);
  - **драйвером сетевой интерфейсной карты** — специальной программой, управляющей работой сетевой интерфейсной карты.
- *Интерфейс компьютер—периферийное устройство* (в данном случае интерфейс компьютер—принтер) позволяет компьютеру управлять работой периферийного устройства (ПУ). Этот интерфейс реализуется:

- со стороны компьютера — **интерфейсной картой** и **драйвером ПУ** (принтера), подобным сетевой интерфейсной карте и ее драйверу;
- со стороны ПУ — **контроллером ПУ** (принтера), обычно представляющим собой аппаратное устройство[1], принимающее от компьютера как *данные*, например байты информации, которую нужно распечатать на бумаге, так и *команды*, которые он отрабатывает, управляя электромеханическими частями периферийного устройства, например выталкивая лист бумаги из принтера или перемещая магнитную головку диска.

Связь компьютера с периферийным устройством

Для того чтобы решить задачу организации доступа приложения, выполняемого на компьютере *A*, к ПУ через сеть, давайте прежде всего посмотрим, как управляет этим устройством приложение, выполняемое на компьютере *B*, к которому данное ПУ подключено непосредственно (см. рис. 2.2).

1. Пусть приложению *B* в какой-то момент потребовалось вывести на печать некоторые данные. Для этого приложение обращается с запросом на выполнение операции ввода-вывода к *операционной системе* (как правило, драйвер не может быть запущен на выполнение непосредственно приложением). В запросе указываются адрес данных, которые необходимо напечатать (адрес буфера ОП), и информация о том, на каком периферийном устройстве эту операцию требуется выполнить.
2. Получив запрос, операционная система запускает программу — *драйвер принтера*. С этого момента все дальнейшие действия по выполнению операции ввода-вывода со стороны компьютера реализуются только драйвером принтера и работающим под его управлением аппаратным модулем — *интерфейсной картой принтера* без участия приложения и операционной системы.
3. Драйвер принтера оперирует командами, понятными контроллеру принтера, такими, например, как «Печать символа», «Перевод строки», «Возврат каретки». Драйвер в определенной последовательности загружает коды этих команд, а также данные, взятые из буфера ОП, в буфер интерфейсной карты принтера, которая побайтно передает их по сети контроллеру принтера.
4. Интерфейсная карта выполняет низкоуровневую работу, не вдаваясь в детали, касающиеся логики управления устройством, смысла данных и команд, передаваемых ей драйвером, считая их однородным потоком байтов. После получения от драйвера очередного байта интерфейсная карта просто последовательно передает биты в линию связи, представляя каждый бит электрическим сигналом. Чтобы контроллеру принтера стало понятно, что начинается передача байта, перед передачей первого бита информационная карта формирует **стартовый сигнал** специфической формы, а после передачи последнего информационного бита — **стоповый сигнал**. Эти сигналы синхронизируют передачу байта. Контроллер, опознав стартовый бит, начинает принимать информационные биты, формируя из них байт в своем приемном буфере. Помимо информационных битов карта может передавать бит контроля четности для повышения достоверности

[1] Встречаются и программно-управляемые контроллеры, например для управления современными принтерами, обладающими сложной логикой.

обмена. При корректно выполненной передаче в буфере принтера устанавливается соответствующий признак.

5. Получив очередной байт, контроллер интерпретирует его и запускает заданную операцию принтера. Закончив работу по печати всех символов документа, драйвер принтера сообщает операционной системе о выполнении запроса, а та, в свою очередь, сигнализирует об этом событии приложению.

Обмен данными между двумя компьютерами

Механизмы взаимодействия компьютеров в сети многое позаимствовали у схемы взаимодействия компьютера с периферийными устройствами. В самом простом случае связь компьютеров может быть реализована с помощью тех же самых средств, которые используются для связи компьютера с периферией, с той разницей, что в этом случае активную роль играют обе взаимодействующие стороны.

Приложения *A* и *B* (см. рис. 2.2) управляют процессом передачи данных путем обмена **сообщениями**. Чтобы приложения могли «понимать» получаемую друг от друга информацию, программисты, разрабатывавшие эти приложения, должны *строго оговорить* форматы и последовательность сообщений, которыми приложения будут обмениваться во время выполнения этой операции. Например, они могут договориться о том, что любая операция обмена данными начинается с передачи сообщения, запрашивающего информацию о готовности приложения *B*; что в следующем сообщении идут идентификаторы компьютера и пользователя, сделавшего запрос; что признаком срочного завершения операции обмена данными является определенная кодовая комбинация и т. п. Тем самым определяется **протокол взаимодействия приложений** для выполнения операции данного типа.

Аналогично тому, как при выводе данных на печать необходимо передавать принтеру дополнительно некоторый объем служебной информации — в виде команд управления принтером, для передачи данных из одного компьютера в другой необходимо сопровождать эти данные дополнительной информацией в виде протокольных сообщений, которыми обмениваются приложения.

Заметим, что для реализации протокола нужно, чтобы к моменту возникновения потребности в обмене данными были активны оба приложения: как приложение *A*, которое посылает инициирующее сообщение, так и приложение *B*, которое должно быть готово принять это сообщение и выработать реакцию на него.

Передача любых данных (как сообщений протокола приложений, так и собственно данных, составляющих цель операции обмена) происходит в соответствии с одной и той же процедурой.

На стороне компьютера A приложение, следуя логике протокола, размещает в буфере ОП либо собственное очередное сообщение, либо данные и обращается к ОС с запросом на выполнение операции межкомпьютерного обмена данными. ОС запускает соответствующий драйвер сетевой карты, который загружает байт из буфера ОП в буфер интерфейсной карты, после чего инициирует ее работу. Сетевая интерфейсная карта последовательно передает биты в линию связи, дополняя каждый новый байт стартовым и стоповым битами.

На стороне компьютера B сетевая интерфейсная карта принимает биты, поступающие со стороны внешнего интерфейса, и помещает их в собственный буфер. После того как получен стоповый бит, интерфейсная карта устанавливает признак завершения приема байта

и выполняет проверку корректности приема, например путем контроля бита четности. Факт корректного приема байта фиксируется драйвером сетевой интерфейсной карты компьютера *B*. Драйвер переписывает принятый байт из буфера интерфейсной карты в заранее зарезервированный буфер ОП компьютера *B*. Приложение *B* извлекает данные из буфера и интерпретирует их в соответствии со своим протоколом либо как сообщение, либо как данные. Если согласно протоколу приложение *B* должно передать ответ приложению *A*, то выполняется симметричная процедура.

Таким образом, связав электрически и информационно два автономно работающих компьютера, мы получили простейшую **компьютерную сеть**.

Доступ к периферийным устройствам через сеть

Итак, мы имеем в своем распоряжении механизм, который позволяет приложениям, выполняющимся на разных компьютерах, обмениваться данными. И хотя приложение *A* (см. рис. 2.2) по-прежнему не может управлять принтером, подключенным к компьютеру *B*, оно может теперь воспользоваться средствами межкомпьютерного обмена данными, чтобы передать приложению *B* «просьбу» выполнить для него требуемую операцию. Приложение *A* должно «объяснить» приложению *B*, какую операцию необходимо выполнить, с какими данными, на каком из имеющихся в его распоряжении устройств, в каком виде должен быть распечатан текст и т. п. В ходе печати могут возникнуть ситуации, о которых приложение *B* должно оповестить приложение *A*, например об отсутствии бумаги в принтере. То есть для решения поставленной задачи — доступа к принтеру по сети — должен быть разработан специальный протокол взаимодействия приложений *A* и *B*.

А теперь посмотрим, как работают вместе все элементы этой простейшей компьютерной сети при решении задачи совместного использования принтера.

1. В соответствии с принятым протоколом приложение *A* формирует сообщение-запрос к приложению *B*, помещает его в буфер ОП компьютера *A* и обращается к ОС, снабжая ее необходимой информацией.
2. ОС запускает драйвер сетевой интерфейсной карты, сообщая ему адрес буфера ОП, где хранится сообщение.
3. Драйвер и сетевая интерфейсная карта компьютера *A*, взаимодействуя с драйвером и интерфейсной картой компьютера *B*, передают сообщение байт за байтом в буфер ОП компьютера *B*.
4. Приложение *B* извлекает сообщение из буфера, интерпретирует его в соответствии с протоколом и выполняет необходимые действия. В число таких действий входит в том числе обращение к ОС с запросом на выполнение тех или иных операций с локальным принтером.
5. ОС запускает драйвер принтера, который в кооперации с интерфейсной картой и контроллером принтера выполняет требуемую операцию печати.

Уже на этом начальном этапе, рассматривая связь компьютера с периферийным устройством, мы столкнулись с важнейшими «сетевыми» понятиями: интерфейсом и протоколом, драйвером и интерфейсной картой, а также с проблемами, характерными для компьютерных сетей: согласованием интерфейсов, синхронизацией асинхронных процессов, обеспечением достоверности передачи данных.

Сетевое программное обеспечение

Мы только что рассмотрели случай совместного использования принтера в простейшей сети, состоящей только из двух компьютеров. Однако даже на этом начальном этапе мы уже можем сделать некоторые выводы относительно строения сетевого программного обеспечения: сетевых служб, сетевой операционной системы и сетевых приложений.

Сетевые службы и сервисы

Потребность в доступе к удаленному принтеру может возникать у пользователей самых разных приложений: текстового редактора, графического редактора, системы управления базой данных (СУБД). Очевидно, что дублирование в каждом из приложений общих для всех них функций по организации удаленной печати является избыточным.

Более эффективным представляется подход, при котором эти функции исключаются из приложений и оформляются в виде пары специализированных программных модулей — *клиента* и *сервера печати* (рис. 2.3), функции которых ранее выполнялись соответственно приложениями *A* и *B*. Теперь эта пара клиент—сервер может быть использована любым приложением, выполняемым на компьютере *A*.

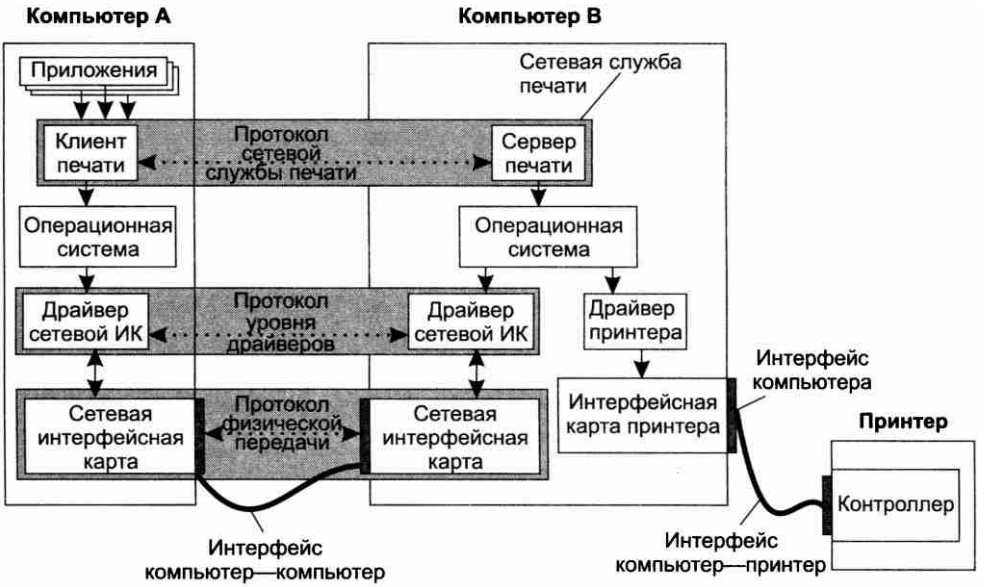


Рис. 2.3. Совместное использование принтера в компьютерной сети с помощью сетевой службы печати

Обобщая такой подход применительно к другим типам разделяемых ресурсов, дадим следующие определения[1].

[1] Сервером и клиентом также называют компьютеры, на которых работают серверная и клиентская части сетевой службы соответственно.

Клиент — это модуль, предназначенный для формирования и передачи сообщений-запросов к ресурсам удаленного компьютера от разных приложений с последующим приемом результатов из сети и передачей их соответствующим приложениям.

Сервер — это модуль, который постоянно ожидает прихода из сети запросов от клиентов и, приняв запрос, пытается его обслужить, как правило, с участием локальной ОС; один сервер может обслуживать запросы сразу нескольких клиентов (поочередно или одновременно).

Пара клиент—сервер, предоставляющая доступ к конкретному типу ресурса компьютера через сеть, образует **сетевую службу**.

Каждая служба связана с определенным типом сетевых ресурсов. Так, на рис. 2.3 модули клиента и сервера, реализующие удаленный доступ к принтеру, образуют сетевую службу печати.

Среди сетевых служб можно выделить такие, которые ориентированы не на простого пользователя, как, например, файловая служба или служба печати, а на администратора. Такие службы направлены на организацию работы сети. Например, **справочная служба**, или **служба каталогов**, предназначена для ведения базы данных о пользователях сети, обо всех ее программных и аппаратных компонентах.

Услуги, предоставляемые службой, называются **сервисом**.

Служба может предоставлять сервис как одного типа, так и нескольких типов. Так, к числу услуг, оказываемых справочной службой, помимо учета ресурсов относятся сервисы аудита, аутентификации, авторизации и др.

Для поиска и просмотра информации в Интернете используется **веб-служба**, состоящая из **веб-сервера** и клиентской программы, называемой **веб-браузером** (web browser). Разделяемым ресурсом в данном случае является **веб-сайт** — определенным образом организованный набор файлов, содержащих связанную в смысловом отношении информацию и хранящихся на внешнем накопителе веб-сервера.

На схеме веб-службы, показанной на рис. 2.4, два компьютера связаны не непосредственно, как это было во всех предыдущих примерах, а через множество промежуточных компьютеров и других сетевых устройств, входящих в состав Интернета. Для того чтобы отразить этот факт графически, мы поместили между двумя компьютерами так называемое **коммуникационное облако**, которое позволяет нам абстрагироваться от всех деталей среды передачи сообщений. Обмен сообщениями между клиентской и серверной частями веб-службы выполняется по стандартному протоколу HTTP и никак не зависит от того, передаются ли эти сообщения «из рук в руки» (от интерфейса одного компьютера к интерфейсу другого) или через большое число посредников — транзитных коммуникационных устройств. Вместе с тем усложнение среды передачи сообщений приводит к возникновению новых дополнительных задач, на решение которых не был рассчитан

упоминавшийся ранее простейший драйвер сетевой интерфейсной карты. Вместо него на взаимодействующих компьютерах должны быть установлены более развитые программные **транспортные средства**.

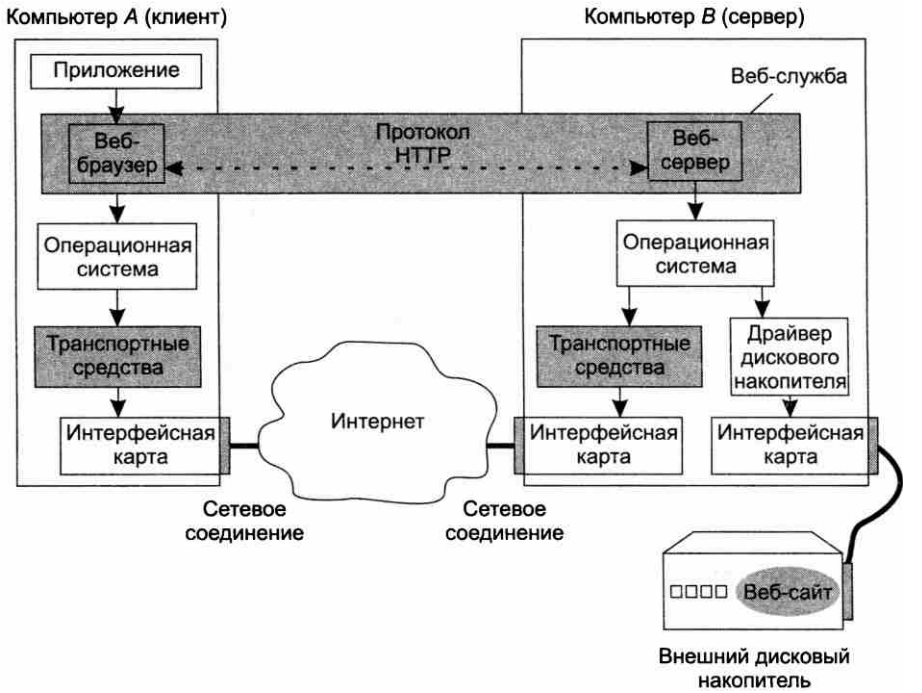


Рис. 2.4. Веб-служба

Сетевая операционная система

Операционную систему компьютера часто определяют как взаимосвязанный набор системных программ, который обеспечивает эффективное управление ресурсами компьютера (памятью, процессором, внешними устройствами, файлами и др.), а также предоставляет пользователю удобный интерфейс для работы с аппаратурой компьютера и разработки приложений.

Говоря о *сетевой* ОС, мы, очевидно, должны расширить границы управляемых ресурсов за пределы одного компьютера.

Сетевой операционной системой называют операционную систему компьютера, которая помимо управления локальными ресурсами предоставляет пользователям и приложениям возможность эффективного и удобного доступа к информационным и аппаратным ресурсам других компьютеров сети.

Сегодня практически все операционные системы являются сетевыми.

Из примеров, рассмотренных в предыдущих разделах (см. рис 2.3 и 2.4), мы видим, что удаленный доступ к сетевым ресурсам обеспечивается:

- сетевыми службами;
- средствами транспортировки сообщений по сети (в простейшем случае — сетевыми интерфейсными картами и их драйверами).

Следовательно, именно эти функциональные модули должны быть добавлены к ОС, чтобы она могла называться сетевой (рис. 2.5).

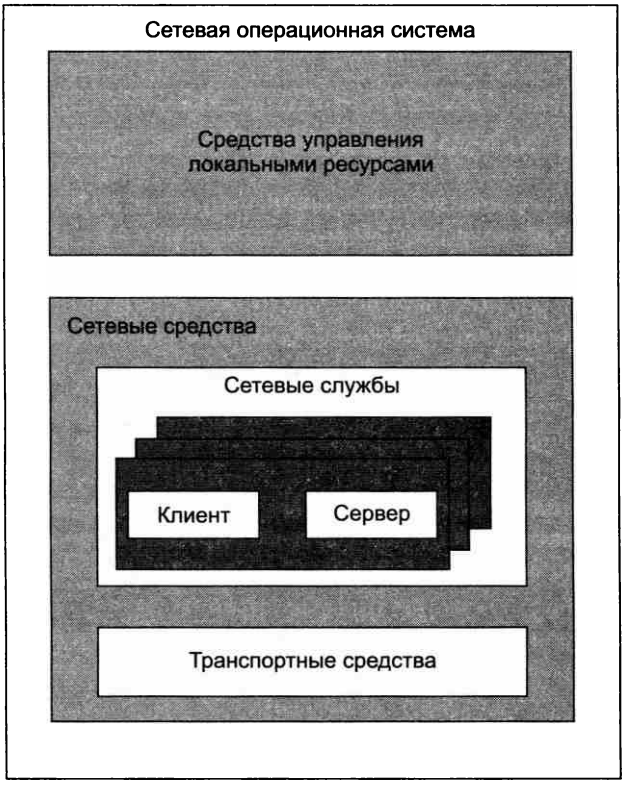


Рис. 2.5. Функциональные компоненты сетевой ОС

От того, насколько богатый набор сетевых служб и услуг предлагает операционная система конечным пользователям, приложениям и администраторам сети, зависит ее позиция в общем ряду сетевых ОС.

Помимо сетевых служб сетевая ОС должна включать *программные коммуникационные* (*транспортные*) *средства*, обеспечивающие совместно с аппаратными коммуникационными средствами передачу сообщений, которыми обмениваются клиентские и серверные части сетевых служб. Задачу коммуникации между компьютерами сети решают драйверы и протокольные модули. Они выполняют такие функции, как формирование сообщений, разбиение сообщения на части (пакеты, кадры), преобразование имен компьютеров

в числовые адреса, дублирование сообщений в случае их потери, определение маршрута в сложной сети и т. д.

И сетевые службы, и транспортные средства могут являться неотъемлемыми (встроенными) компонентами ОС или существовать в виде отдельных программных продуктов. Например, сетевая файловая служба обычно встраивается в ОС, а вот веб-браузер чаще всего приобретается отдельно. Типичная сетевая ОС имеет в своем составе широкий набор драйверов и протокольных модулей, однако у пользователя, как правило, есть возможность дополнить этот стандартный набор необходимыми ему программами. Решение о способе реализации клиентов и серверов сетевой службы, а также драйверов и протокольных модулей принимается разработчиками с учетом самых разных соображений: технических, коммерческих и даже юридических. Так, например, именно на основании антимонопольного закона США компании Microsoft было запрещено включать ее браузер Internet Explorer в состав ОС этой компании.

Сетевая служба может быть представлена в ОС либо обеими (клиентской и серверной) частями, либо только одной из них.

В первом случае операционная система, называемая **одноранговой**, не только позволяет обращаться к ресурсам других компьютеров, но и предоставляет собственные ресурсы в распоряжение пользователей других компьютеров. Например, если на всех компьютерах сети установлены и клиенты, и серверы файловой службы, то все пользователи сети могут совместно использовать файлы друг друга. Компьютеры, совмещающие функции клиента и сервера, называют одноранговыми узлами.

Операционная система, которая преимущественно содержит клиентские части сетевых служб, называется **клиентской**. Клиентские ОС устанавливаются на компьютеры, обращающиеся с запросами к ресурсам других компьютеров сети. За такими компьютерами, также называемыми клиентскими, работают рядовые пользователи. Обычно клиентские компьютеры относятся к классу относительно простых устройств.

К другому типу операционных систем относится **серверная ОС** — она ориентирована на обработку запросов из сети к ресурсам своего компьютера и включает в себя в основном серверные части сетевых служб. Компьютер с установленной на нем серверной ОС, занимающийся исключительно обслуживанием запросов других компьютеров, называют **выделенным сервером** сети. За выделенным сервером, как правило, обычные пользователи не работают.

ПРИМЕЧАНИЕ

Подробнее о сетевых операционных системах и встроенных в них сетевых службах вы можете прочитать в специальной литературе, а также в учебнике авторов «Сетевые операционные системы». Наиболее популярные сетевые службы Интернета, такие как электронная почта, веб-служба, IP-телефония и другие, рассматриваются далее в части VI книги.

Сетевые приложения

На компьютере, подключенном к сети, могут запускаться приложения нескольких типов:

- **Локальное приложение** целиком выполняется на данном компьютере и использует только локальные ресурсы (рис. 2.6, *а*). Для такого приложения не требуется никаких сетевых средств, оно может быть выполнено на автономно работающем компьютере.

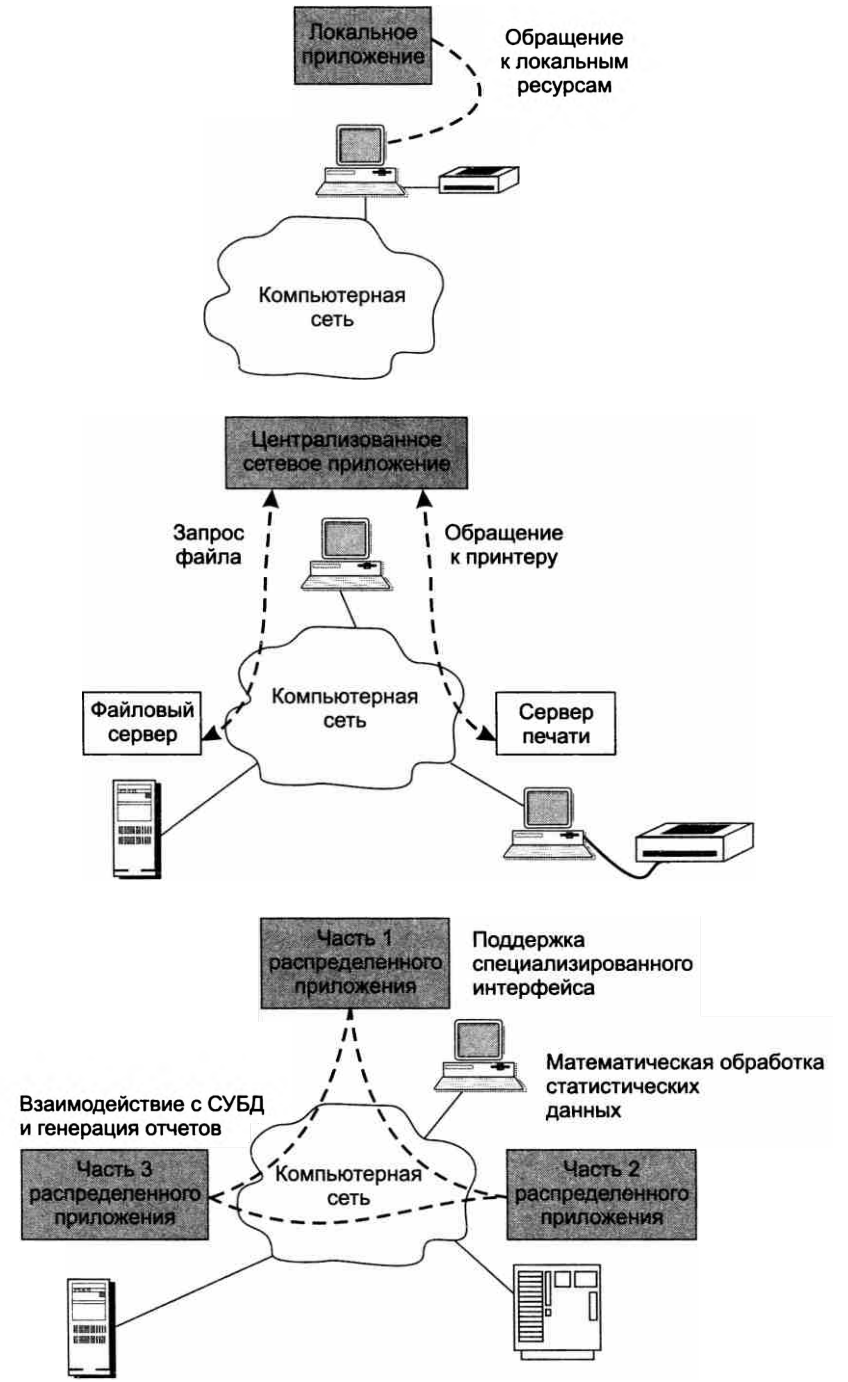


Рис. 2.6. Типы приложений, выполняющихся в сети

- **Централизованное сетевое приложение** целиком выполняется на данном компьютере, но обращается в процессе своей работы к ресурсам других компьютеров сети. В примере на рис. 2.6, *б* приложение, которое выполняется на клиентском компьютере, обрабатывает данные из файла, хранящегося на файл-сервере, а затем распечатывает результаты на принтере, подключенном к серверу печати. Очевидно, что работа такого типа приложений невозможна без участия сетевых служб и средств транспортировки сообщений.
- **Распределенное (сетевое) приложение** состоит из нескольких взаимодействующих частей, каждая из которых выполняет какую-то определенную законченную работу по решению прикладной задачи, причем каждая часть может выполняться и, как правило, выполняется на отдельном компьютере сети (рис. 2.6, *в*). Части распределенного приложения взаимодействуют друг с другом, используя сетевые службы и транспортные средства ОС. Распределенное приложение в общем случае имеет доступ ко всем ресурсам компьютерной сети.

Очевидным преимуществом распределенных приложений является возможность распараллеливания вычислений, а также специализация компьютеров. Так, в приложении, предназначенном, скажем, для анализа климатических изменений, можно выделить три достаточно самостоятельные части (см. рис. 2.6, *в*), допускающие распараллеливание. Первая часть приложения, выполняющаяся на сравнительно маломощном персональном компьютере, могла бы поддерживать специализированный графический пользовательский интерфейс, вторая — заниматься статистической обработкой данных на высокопроизводительном мэйнфрейме, третья — генерировать отчеты на сервере с установленной стандартной СУБД. В общем случае каждая из частей распределенного приложения может быть представлена несколькими копиями, работающими на разных компьютерах. Скажем, в данном примере первую часть, ответственную за поддержку специализированного пользовательского интерфейса, можно было бы запустить на нескольких персональных компьютерах, что позволило бы работать с этим приложением нескольким пользователям одновременно.

Однако чтобы добиться всех тех преимуществ, которые сулят распределенные приложения, разработчикам этих приложений приходится решать множество проблем, например: на сколько частей следует разбить приложение, какие функции возложить на каждую часть, как организовать взаимодействие этих частей, чтобы в случае сбоев и отказов оставшиеся части корректно завершали работу, и т. д. и т. п.

Заметим, что все сетевые службы, включая файловую службу, службу печати, службу электронной почты, службу удаленного доступа, интернет-телефонию и т. д., по определению относятся к классу распределенных приложений. Действительно, любая сетевая служба включает в себя клиентскую и серверную части, которые могут выполняться и обычно выполняются на разных компьютерах.

На рис. 2.7, иллюстрирующем распределенный характер веб-службы, мы видим различные виды клиентских устройств — персональные компьютеры, ноутбуки и мобильные телефоны — с установленными на них веб-браузерами, которые взаимодействуют по сети с веб-сервером. Таким образом, с одним и тем же веб-сайтом может одновременно работать множество — сотни и тысячи — сетевых пользователей.

Многочисленные примеры распределенных приложений можно встретить и в такой области, как обработка данных научных экспериментов. Это неудивительно, так как многие

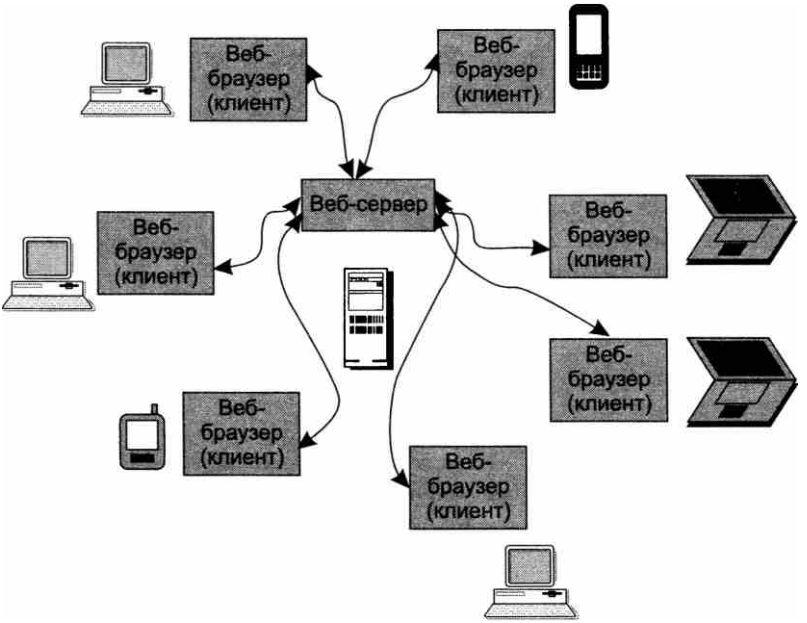


Рис. 2.7. Веб-служба как распределенное приложение

эксперименты порождают такие большие объемы данных, генерируемых в реальном масштабе времени, которые просто невозможно обработать на одном, даже очень мощном, суперкомпьютере. Кроме того, алгоритмы обработки экспериментальных данных часто легко распараллеливаются, что также важно для успешного применения взаимосвязанных компьютеров с целью решения какой-либо общей задачи. Одним из известных примеров распределенного научного приложения является программное обеспечение обработки данных большого адронного коллайдера (Large Hadron Collider, LHC), запущенного 10 сентября 2008 года в CERN, — это приложение работает более чем на 30 тысячах компьютеров, объединенных в сеть.

Физическая передача данных по линиям связи

Даже при рассмотрении простейшей сети, состоящей всего из двух машин, можно выявить многие проблемы, связанные с физической передачей сигналов по линиям связи.

Кодирование

В вычислительной технике для представления данных используется **двоичный код**. Внутри компьютера единицам и нулям данных соответствуют дискретные электрические сигналы.

Представление данных в виде электрических или оптических сигналов называется **кодированием**.

Существуют различные способы кодирования двоичных цифр, например **потенциальный способ**, при котором единице соответствует один уровень напряжения, а нулю — другой, или **импульсный способ**, когда для представления цифр используются импульсы различной полярности.

Аналогичные подходы применимы для кодирования данных и при передаче их между двумя компьютерами **по линиям связи**. Однако эти линии связи отличаются по своим характеристикам от линий внутри компьютера. Главное отличие внешних линий связи от внутренних состоит в их гораздо большей протяженности, а также в том, что они проходят вне экранированного корпуса по пространствам, зачастую подверженным воздействию сильных электромагнитных помех. Все это приводит к существенно бóльшим искажениям прямоугольных импульсов (например, «заваливанию» фронтов), чем внутри компьютера. Поэтому для надежного распознавания импульсов на приемном конце линии связи при передаче данных внутри и вне компьютера не всегда можно использовать одни и те же скорости и способы кодирования. Например, медленное нарастание фронта импульса из-за высокой емкостной нагрузки линии требует, чтобы импульсы передавались с меньшей скоростью (чтобы передний и задний фронты соседних импульсов не перекрывались и импульс успел «дорасти» до требуемого уровня).

В вычислительных сетях применяют как потенциальное, так и импульсное кодирование дискретных данных, а также специфический способ представления данных, который никогда не используется внутри компьютера, — **модуляцию** (рис. 2.8). При модуляции дискретная информация представляется синусоидальным сигналом той частоты, которую хорошо передает имеющаяся линия связи.

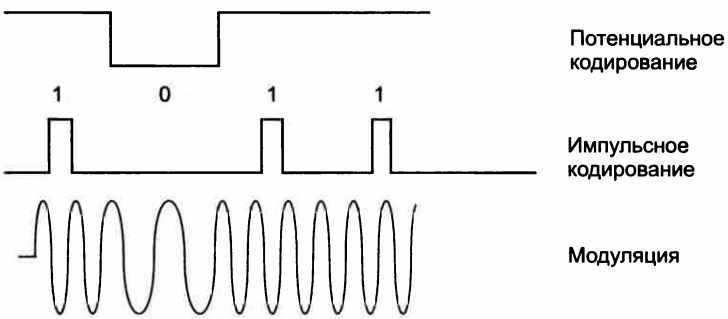


Рис. 2.8. Примеры представления дискретной информации

Потенциальное и импульсное кодирование применяется на каналах *высокого качества*, а модуляция на основе синусоидальных сигналов предпочтительнее в том случае, когда канал вносит сильные искажения в передаваемые сигналы. Например, модуляция используется в глобальных сетях при передаче данных через аналоговые телефонные каналы связи, которые были разработаны для передачи голоса в аналоговой форме и поэтому плохо подходят для непосредственной передачи импульсов.

На способ передачи сигналов влияет и *количество проводов* в линиях связи между компьютерами. Для снижения стоимости линий связи в сетях обычно стремятся к сокращению количества проводов и из-за этого передают все биты одного байта или даже нескольких байтов не параллельно, как это делается внутри компьютера, а последовательно (побитно), для чего достаточно всего одной пары проводов.

Еще одной проблемой, которую нужно решать при передаче сигналов, является проблема взаимной **синхронизации** передатчика одного компьютера с приемником другого. При организации взаимодействия модулей внутри компьютера эта проблема решается очень просто, так как в этом случае все модули синхронизируются от общего тактового генератора. Проблема синхронизации при связи компьютеров может решаться разными способами, как путем обмена специальными тактовыми синхроимпульсами по отдельной линии, так и путем периодической синхронизации заранее обусловленными кодами или импульсами характерной формы, отличающейся от формы импульсов данных.

Несмотря на предпринимаемые меры (выбор соответствующей скорости обмена данными, линий связи с определенными характеристиками, способа синхронизации приемника и передатчика), существует вероятность искажения некоторых битов передаваемых данных. Для повышения надежности передачи данных между компьютерами, как правило, используется стандартный прием — подсчет **контрольной суммы** и передача полученного значения по линиям связи после каждого байта или после некоторого блока байтов. Часто в протокол обмена данными включается как обязательный элемент **сигнал-квитанция**, который подтверждает правильность приема данных и посылается от получателя отправителю.

Характеристики физических каналов

Существует большое количество характеристик, связанных с передачей трафика через физические каналы. С теми из них, которые будут необходимы нам уже в ближайшее время, мы коротко познакомимся сейчас, а позже изучим их и некоторые другие сетевые характеристики более детально[1].

- **Предложенная нагрузка** — это поток данных, поступающий от пользователя на вход сети. Предложенную нагрузку можно характеризовать скоростью поступления данных в сеть в битах в секунду (или килобитах, мегабитах и т. д.).
- **Скорость передачи данных** (information rate, или throughput, оба английских термина используются равноправно) — это *фактическая* скорость потока данных, прошедшего через сеть. Эта скорость может быть меньше, чем скорость предложенной нагрузки, так как данные в сети могут искажаться или теряться.
- **Емкость канала связи** (capacity), называемая также **пропускной способностью**, представляет собой *максимально возможную* скорость передачи информации по каналу.

Спецификой этой характеристики является то, что она отражает не только параметры *физической среды передачи*, но и особенности *выбранного способа передачи* дискретной информации в этой среде. Например, емкость канала связи в сети Ethernet на оптическом волокне равна 10 Мбит/с. Эта скорость является предельно возможной для сочетания технологии Ethernet и оптического волокна. Однако для того же самого оптического

[1] См. главу 5, а также раздел «Характеристики линий связи» в главе 7.

волокна можно разработать другую технологию передачи, отличающуюся способом кодирования данных, тактовой частотой и другими параметрами, которая будет иметь другую емкость. Так, технология Fast Ethernet обеспечивает передачу данных по тому же оптическому волокну с максимальной скоростью 100 Мбит/с, а технология Gigabit Ethernet — 1000 Мбит/с. Передатчик коммуникационного устройства должен работать со скоростью, равной пропускной способности канала. Эта скорость иногда называется **битовой скоростью передатчика** (bit rate of transmitter).

- **Полоса пропускания** (bandwidth) — этот термин может ввести в заблуждение, потому что он используется в двух разных значениях. Во-первых, с его помощью могут характеризовать *среду передачи*. В этом случае он означает ширину полосы частот, которую линия передает без существенных искажений. Из этого определения понятно происхождение термина. Во-вторых, термин «полоса пропускания» используется как синоним термина *емкость канала связи*. В первом случае полоса пропускания измеряется в герцах (Гц), во втором — в битах в секунду. Различать значения термина нужно по контексту, хотя иногда это достаточно трудно. Конечно, лучше было бы применять разные термины для описания различных характеристик, но существуют традиции, которые изменить трудно. Такое двойное использование термина «полоса пропускания» уже вошло во многие стандарты и книги, поэтому и в данной книге мы будем следовать сложившемуся подходу. Нужно также учитывать, что этот термин в его втором значении является даже более распространенным, чем емкость, поэтому из этих двух синонимов мы будем использовать полосу пропускания.

Еще одна группа характеристик канала связи связана с возможностью передачи информации по каналу в одну или обе стороны.

При взаимодействии двух компьютеров обычно требуется передавать информацию в обоих направлениях, от компьютера *A* к компьютеру *B* и обратно. Даже в том случае, когда пользователю кажется, что он только получает информацию (например, загружает музыкальный файл из Интернета) или только ее передает (отправляет электронное письмо), обмен информации идет в двух направлениях. Просто существует основной поток данных, которые интересуют пользователя, и вспомогательный поток противоположного направления, который образуют квитанции о получении этих данных.

Физические каналы связи делятся на несколько типов в зависимости от того, могут они передавать информацию в обоих направлениях или нет.

- **Дуплексный канал** обеспечивает одновременную передачу информации в обоих направлениях. Дуплексный канал может состоять их двух физических сред, каждая их которых используется для передачи информации только в одном направлении. Возможен вариант, когда одна среда служит для одновременной передачи встречных потоков, в этом случае применяют дополнительные методы выделения каждого потока из суммарного сигнала.
- **Полудуплексный канал** также обеспечивает передачу информации в обоих направлениях, но не одновременно, а по очереди. То есть в течение определенного периода времени информация передается в одном направлении, а в течение следующего периода — в обратном.
- **Симплексный канал** позволяет передавать информацию только в одном направлении. Часто дуплексный канал состоит из двух симплексных каналов.

Подробно вопросы физической передачи дискретных данных обсуждаются в части II.

Проблемы связи нескольких компьютеров

До сих пор мы рассматривали вырожденную сеть, состоящую всего из двух машин. При объединении в сеть большего числа компьютеров возникает целый комплекс новых проблем.

Топология физических связей

Объединяя в сеть несколько (больше двух) компьютеров, необходимо решить, каким образом соединить их друг с другом, другими словами, выбрать конфигурацию физических связей, или топологию.

Под **топологией сети** понимается конфигурация графа, вершинам которого соответствуют конечные узлы сети (например, компьютеры) и коммуникационное оборудование (например, маршрутизаторы), а ребрам — физические или информационные связи между вершинами.

Число возможных вариантов конфигурации резко возрастает при увеличении числа связываемых устройств. Так, если три компьютера мы можем связать двумя способами (рис. 2.9, *а*), то для четырех можно предложить уже шесть топологически разных конфигураций (при условии неразличимости компьютеров), что и иллюстрирует рис. 2.9, *б*.

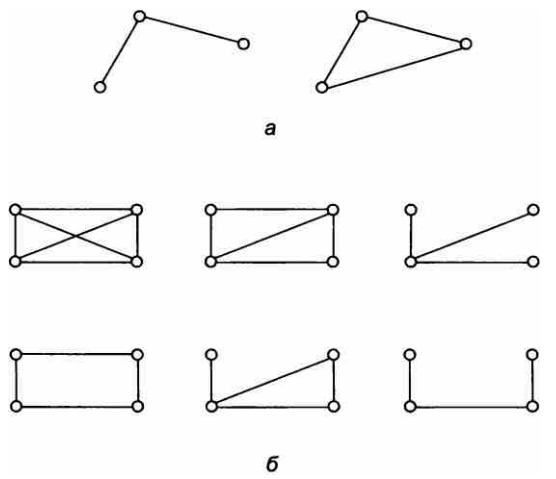


Рис. 2.9. Варианты связи компьютеров

Мы можем соединять каждый компьютер с каждым или же связывать их последовательно, предполагая, что они будут общаться, передавая сообщения друг другу «транзитом». Транзитные узлы должны быть оснащены специальными средствами, позволяющими им выполнять эту специфическую посредническую операцию. В качестве транзитного узла может выступать как универсальный компьютер, так и специализированное устройство.

От выбора топологии связей существенно зависят характеристики сети. Например, наличие между узлами нескольких путей повышает надежность сети и делает возможным распределение загрузки между отдельными каналами. Простота присоединения новых узлов, свойственная некоторым топологиям, делает сеть легко *расширяемой*. Экономические соображения часто приводят к выбору топологий, для которых характерна минимальная суммарная длина линий связи.

Среди множества возможных конфигураций различают полносвязные и неполносвязные.

Полносвязная топология соответствует сети, в которой каждый компьютер непосредственно связан со всеми остальными (рис. 2.10, *а*). Несмотря на логическую простоту, этот вариант оказывается громоздким и неэффективным. Действительно, в таком случае каждый компьютер в сети должен иметь большое количество коммуникационных портов, достаточное для связи с каждым из остальных компьютеров сети. Для каждой пары компьютеров должна быть выделена отдельная физическая линия связи. (В некоторых случаях даже две, если невозможно использование этой линии для двусторонней передачи.) Полносвязные топологии в крупных сетях применяются редко, так как для связи N узлов требуется $N(N-1)/2$ физических дуплексных линий связей, то есть имеет место квадратичная зависимость от числа узлов. Чаще этот вид топологии используется в многомашинных комплексах или в сетях, объединяющих небольшое количество компьютеров.

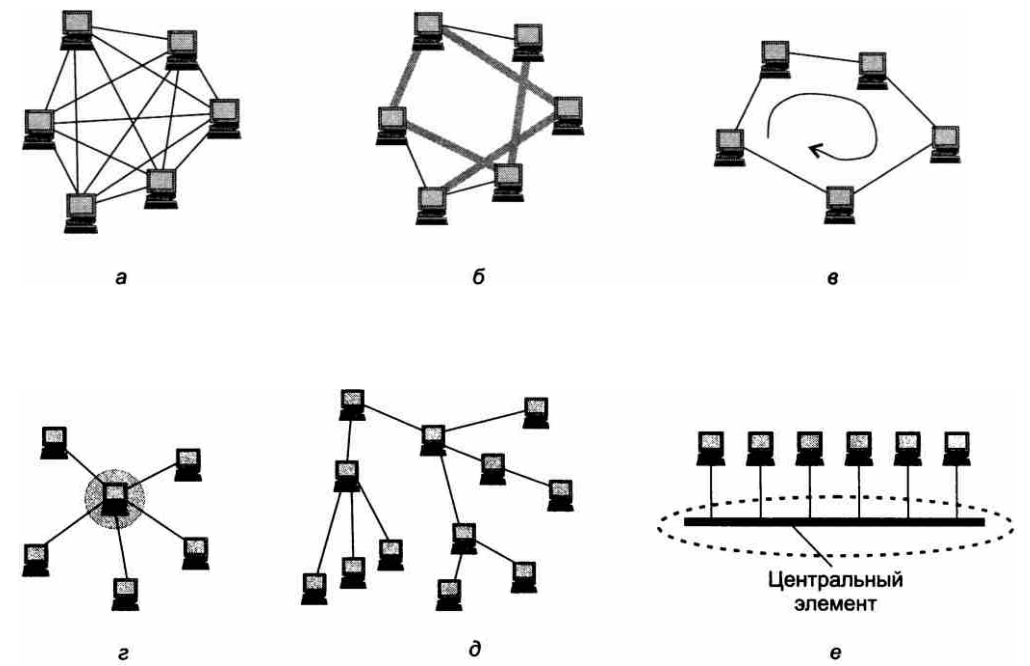


Рис. 2.10. Типовые топологии сетей

Все другие варианты основаны на **неполносвязных топологиях**, когда для обмена данными между двумя компьютерами может потребоваться транзитная передача данных через другие узлы сети.

Ячеистая топология[1] получается из полносвязной путем удаления некоторых связей (рис. 2.10, *б*). Ячеистая топология допускает соединение большого количества компьютеров и характерна, как правило, для крупных сетей.

В сетях с **кольцевой топологией** (рис. 2.10, *в*) данные передаются по кольцу от одного компьютера к другому. Главным достоинством кольца является то, что оно по своей природе обеспечивает резервирование связей. Действительно, любая пара узлов соединена здесь двумя путями — по часовой стрелке и против нее. Кроме того, кольцо представляет собой очень удобную конфигурацию для организации обратной связи — данные, сделав полный оборот, возвращаются к узлу-источнику. Поэтому источник может контролировать процесс доставки данных адресату. Часто это свойство кольца используется для тестирования связности сети и поиска узла, работающего некорректно. В то же время в сетях с кольцевой топологией необходимо принимать специальные меры, чтобы в случае выхода из строя или отключения какого-либо компьютера не прерывался канал связи между остальными узлами кольца.

Звездообразная топология (рис. 2.10, *г*) образуется в случае, когда каждый компьютер подключается непосредственно к общему центральному устройству, называемому **концентратором**[2]. В функции концентратора входит направление передаваемой компьютером информации одному или всем остальным компьютерам сети. В качестве концентратора может выступать как универсальный компьютер, так и специализированное устройство. К недостаткам звездообразной топологии относится более высокая стоимость сетевого оборудования из-за необходимости приобретения специализированного центрального устройства. Кроме того, возможности по наращиванию количества узлов в сети ограничиваются количеством портов концентратора.

Иногда имеет смысл строить сеть с использованием нескольких концентраторов, иерархически соединенных между собой звездообразными связями (рис. 2.10, *д*). Получаемую в результате структуру называют **иерархической звездой**, или **деревом**. В настоящее время дерево является самой распространенной топологией связей как в локальных, так и глобальных сетях.

Особым частным случаем звезды является **общая шина** (рис. 2.10, *е*). Здесь в качестве центрального элемента выступает пассивный кабель, к которому по схеме «монтажного ИЛИ» подключается несколько компьютеров (такую же топологию имеют многие сети, использующие беспроводную связь, — роль общей шины здесь играет общая радиосреда). Передаваемая информация распространяется по кабелю и доступна одновременно всем компьютерам, присоединенным к этому кабелю. Основными преимуществами такой схемы являются ее дешевизна и простота присоединения новых узлов к сети, а недостатками — низкая надежность (любой дефект кабеля полностью парализует всю сеть) и невысокая производительность (в каждый момент времени только один компьютер может передавать данные по сети, поэтому пропускная способность делится здесь между всеми узлами сети).

В то время как небольшие сети, как правило, имеют типовую топологию — звезда, кольцо или общая шина, для крупных сетей характерно наличие произвольных связей между компьютерами. В таких сетях можно выделить отдельные произвольно связанные фрагменты

---

[1] Иногда ячеистой называют полносвязную или близкую к полносвязной топологию.

[2] В данном случае термин «концентратор» используется в широком смысле, обозначая любое многовходовое устройство, способное служить центральным элементом, например коммутатор или маршрутизатор.

(подсети), имеющие типовую топологию, поэтому их называют сетями со **смешанной топологией** (рис. 2.11).

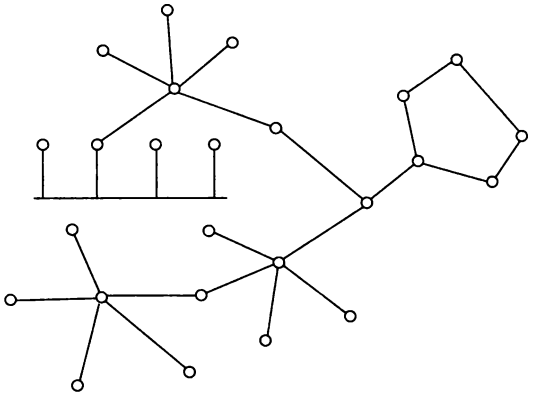


Рис. 2.11. Смешанная топология

Адресация узлов сети

Еще одной новой проблемой, которую нужно учитывать при объединении трех и более компьютеров, является проблема их адресации, точнее, адресации их сетевых интерфейсов[1]. Один компьютер может иметь несколько сетевых интерфейсов. Например, для создания полносвязной структуры из N компьютеров необходимо, чтобы у каждого из них имелся $N - 1$ интерфейс.

По количеству адресуемых интерфейсов адреса можно классифицировать следующим образом:

- **уникальный адрес** (unicast) используется для идентификации отдельных интерфейсов;
- **групповой адрес** (multicast) идентифицирует сразу несколько интерфейсов, поэтому данные, помеченные групповым адресом, доставляются каждому из узлов, входящих в группу;
- данные, направленные по **широковещательному адресу** (broadcast), должны быть доставлены всем узлам сети;
- **адрес произвольной рассылки** (anycast), определенный в новой версии протокола IPv6, так же как и групповой адрес, задает группу адресов, однако данные, посланные по этому адресу, доставляются не всем узлам данной группы, а только одному из них. Выбор этого узла осуществляется в соответствии с некоторыми правилами предпочтения.

Адреса могут быть **числовыми** (например, 129.26.255.255 или 81.1a.ff.ff) и **символьными** (site.domen.ru, willi-winki).

Символьные адреса (имена) предназначены для запоминания людьми и поэтому обычно несут смысловую нагрузку. Для работы в больших сетях символьное имя может иметь иерархическую структуру, например ftp-arch1.ucl.ac.uk. Этот адрес говорит о том, что

---

[1] Иногда вместо точного выражения «адрес сетевого интерфейса» мы будем использовать упрощенное — «адрес узла сети».

данный компьютер поддерживает ftp-архив в сети одного из колледжей Лондонского университета (University College London — ucl) и эта сеть относится к академической ветви (ac) Интернета Великобритании (United Kingdom — uk). При работе в пределах сети Лондонского университета такое длинное символьное имя явно избыточно, и вместо него можно пользоваться кратким символьным именем ftp-arch1. Хотя символьные имена удобны для людей, из-за переменного формата и потенциально большой длины их передача по сети не экономична.

Множество всех адресов, которые являются допустимыми в рамках некоторой схемы адресации, называется **адресным пространством**. Адресное пространство может иметь плоскую (линейную) или иерархическую организацию.

При **плоской организации** множество адресов никак не структурировано. Примером плоского числового адреса является **MAC-адрес**, предназначенный для однозначной идентификации сетевых интерфейсов в локальных сетях. Такой адрес обычно используется только аппаратурой, поэтому его стараются сделать по возможности компактным и записывают в виде двоичного или шестнадцатеричного числа, например 0081005e24a8. При задании MAC-адресов не требуется выполнять никакой ручной работы, так как они обычно встраиваются в аппаратуру компанией-изготовителем, поэтому их называют также **аппаратными адресами** (hardware address). Использование плоских адресов является жестким решением — при замене аппаратуры, например сетевого адаптера, изменяется и адрес сетевого интерфейса компьютера.

При **иерархической** организации адресное пространство структурируется в виде вложенных друг в друга подгрупп, которые, последовательно сужая адресуемую область, в конце концов, определяют отдельный сетевой интерфейс. Например, в трехуровневой структуре адресного пространства адрес конечного узла может задаваться тремя составляющими:

- идентификатором группы (K), в которую входит данный узел;
- идентификатором подгруппы (L);
- идентификатором узла (n), однозначно определяющим его в подгруппе.

Иерархическая адресация во многих случаях оказывается более рациональной, чем плоская. В больших сетях, состоящих из многих тысяч узлов, использование плоских адресов приводит к большим издержкам — конечным узлам и коммуникационному оборудованию приходится оперировать таблицами адресов, состоящими из тысяч записей. В противоположность этому иерархическая система адресации позволяет при перемещении данных до определенного момента пользоваться только старшей составляющей адреса (например, идентификатором группы K), затем для дальнейшей локализации адресата задействовать следующую по старшинству часть (L) и в конечном счете — младшую часть (n).

Типичными представителями иерархических числовых адресов являются сетевые IP-адреса. В них поддерживается двухуровневая иерархия, адрес делится на старшую часть — номер сети и младшую — номер узла. Такое деление позволяет передавать сообщения между сетями только на основании номера сети, а номер узла требуется уже после доставки сообщения в нужную сеть, точно так же, как название улицы используется почтальоном только после того, как письмо доставлено в нужный город.

На практике обычно применяют сразу несколько схем адресации, так что сетевой интерфейс компьютера может одновременно иметь несколько адресов-имен. Каждый адрес

задействуется в той ситуации, когда соответствующий вид адресации наиболее удобен. А для преобразования адресов из одного вида в другой используются специальные вспомогательные протоколы, которые называют **протоколами разрешения адресов**.

Пользователи адресуют компьютеры иерархическими символьными именами, которые автоматически заменяются в сообщениях, передаваемых по сети, иерархическими числовыми адресами. С помощью этих числовых адресов сообщения доставляются из одной сети в другую, а после доставки сообщения в сеть назначения вместо иерархического числового адреса используется плоский аппаратный адрес компьютера. Проблема установления соответствия между адресами различных типов может решаться как централизованными, так и распределенными средствами.

При *централизованном подходе* в сети выделяется один или несколько компьютеров (серверов имен), в которых хранится таблица соответствия имен различных типов, например символьных имен и числовых адресов. Все остальные компьютеры обращаются к серверу имен с запросами, чтобы по символьному имени найти числовой номер необходимого компьютера.

При *распределенном подходе* каждый компьютер сам хранит все назначенные ему адреса разного типа. Тогда компьютер, которому необходимо определить по известному иерархическому числовому адресу некоторого компьютера его плоский аппаратный адрес, посылает в сеть широковещательный запрос. Все компьютеры сети сравнивают содержащийся в запросе адрес с собственным. Тот компьютер, у которого обнаружилось совпадение, посылает ответ, содержащий искомый аппаратный адрес. Такая схема использована в **протоколе разрешения адресов** (Address Resolution Protocol, ARP) стека TCP/IP.

Достоинство распределенного подхода состоит в том, что он позволяет отказаться от выделения специального компьютера в качестве сервера имен, который к тому же часто требует ручного задания таблицы соответствия адресов. Недостатком его является необходимость широковещательных сообщений, перегружающих сеть. Именно поэтому распределенный подход используется в небольших сетях, а централизованный — в больших.

До сих пор мы говорили об адресах сетевых интерфейсов, компьютеров и коммуникационных устройств, однако конечной целью данных, пересылаемых по сети, являются не сетевые интерфейсы или компьютеры, а выполняемые на этих устройствах программы — процессы. Поэтому в адресе назначения наряду с информацией, идентифицирующей интерфейс устройства, должен указываться адрес процесса, которому предназначены посылаемые по сети данные. Очевидно, что достаточно обеспечить уникальность адреса процесса в пределах компьютера. Примером адресов процессов являются *номера портов TCP и UDP*, используемые в стеке TCP/IP.

Коммутация

Пусть компьютеры физически связаны между собой в соответствии с некоторой топологией и выбрана система адресации. Остается нерешенным вопрос: каким образом передавать данные между конечными узлами? Особую сложность приобретает эта задача для неполносвязной топологии сети, когда обмен данными между произвольной парой конечных узлов (пользователей) должен идти в общем случае через транзитные узлы.

Соединение конечных узлов через сеть транзитных узлов называют **коммутацией**. Последовательность узлов, лежащих на пути от отправителя к получателю, образует **маршрут**.

Например, в сети, показанной на рис. 2.12, узлы 2 и 4, непосредственно между собой не связанные, вынуждены передавать данные через транзитные узлы, в качестве которых могут выступить, например, узлы 1 и 5. Узел 1 должен выполнить передачу данных между своими интерфейсами *A* и *B*, а узел 5 — между интерфейсами *F* и *B*. В данном случае маршрутом является последовательность: 2-1-5-4, где 2 — узел-отправитель, 1 и 5 — транзитные узлы, 4 — узел-получатель.

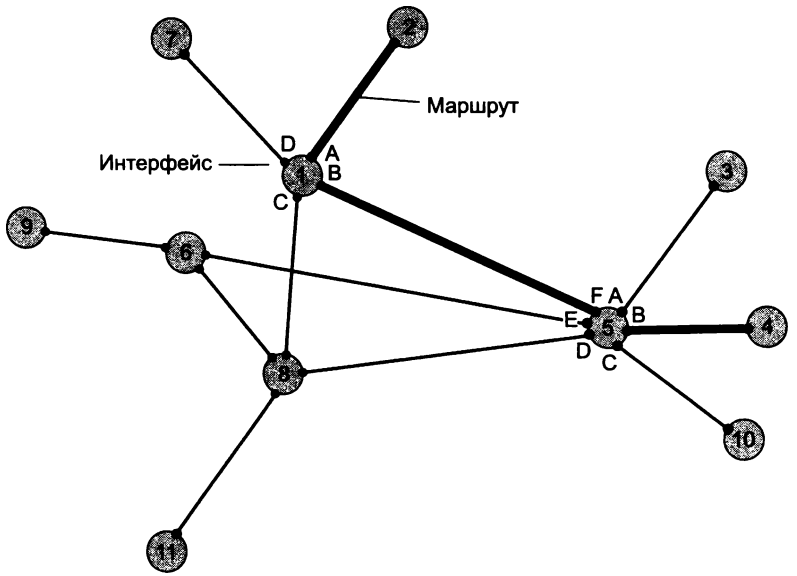


Рис. 2.12. Коммутация абонентов через сеть транзитных узлов

Обобщенная задача коммутации

В самом общем виде задача коммутации может быть представлена в виде следующих взаимосвязанных частных задач.

1. Определение информационных потоков, для которых требуется прокладывать маршруты.
2. Маршрутизация потоков.
3. Продвижение потоков, то есть распознавание потоков и их локальная коммутация на каждом транзитном узле.
4. Мультиплексирование и демультиплексирование потоков.

Определение информационных потоков

Понятно, что через один транзитный узел может проходить несколько маршрутов, например через узел 5 (см. рис. 2.12) проходят как минимум все данные, направляемые узлом 4 каждому из остальных узлов, а также все данные, поступающие в узлы 3, 4 и 10. Транзитный узел должен уметь *распознавать* поступающие на него потоки данных, чтобы обеспе-

чивать передачу каждого из них именно на тот свой интерфейс, который ведет к нужному узлу, и, возможно, чтобы выбрать специфический для данного потока способ его обработки.

Информационным потоком, или потоком данных, называют непрерывную последовательность данных, объединенных набором общих признаков, выделяющих эти данные из общего сетевого трафика.

Например, как поток можно определить все данные, поступающие от одного компьютера; объединяющим признаком в данном случае служит адрес источника. Эти же данные можно представить как совокупность нескольких **подпотоков**, каждый из которых в качестве дифференцирующего признака имеет адрес назначения. Наконец, каждый из этих подпотоков, в свою очередь, можно разделить на более мелкие подпотоки, порожденные разными сетевыми приложениями — электронной почтой, программой копирования файлов, веб-сервером. Данные, образующие поток, могут быть представлены в виде различных информационных единиц данных — пакетов, кадров, ячеек.

ПРИМЕЧАНИЕ

В англоязычной литературе для потоков данных, передающихся с равномерной и неравномерной скоростью, обычно используют разные термины — соответственно «data stream» и «data flow». Например, при передаче веб-страницы через Интернет предложенная нагрузка представляет собой неравномерный поток данных, а при вещании музыки интернет-станцией — равномерный. Для сетей передачи данных характерна неравномерная скорость передачи, поэтому далее в большинстве ситуаций под термином «поток данных» мы будем понимать именно неравномерный поток данных и указывать на равномерный характер этого процесса только тогда, когда это нужно подчеркнуть.

Очевидно, что при коммутации в качестве обязательного признака выступает **адрес назначения** данных. На основании этого признака весь поток входящих в транзитный узел данных разделяется на подпотоки, каждый из которых передается на интерфейс, соответствующий маршруту продвижения данных.

Адреса источника и назначения определяют поток для пары соответствующих конечных узлов. Однако часто бывает полезно представить этот поток в виде нескольких подпотоков, причем для каждого из них может быть проложен свой особый маршрут. Рассмотрим пример, когда на одной и той же паре конечных узлов выполняется несколько взаимодействующих по сети приложений, каждое из которых предъявляет к сети свои особые требования. В таком случае выбор маршрута должен осуществляться с учетом характера передаваемых данных, например для файлового сервера важно, чтобы передаваемые им большие объемы данных направлялись по каналам, обладающим высокой пропускной способностью, а для программной системы управления, которая посылает в сеть короткие сообщения, требующие обязательной и немедленной отработки, при выборе маршрута более важна надежность линии связи и минимальный уровень задержек на маршруте. Кроме того, даже для данных, предъявляющих к сети одинаковые требования, может прокладываться несколько маршрутов, чтобы за счет распараллеливания ускорить передачу данных.

Возможна и обратная по отношению к выделению подпотоков операция — *агрегирование потоков*. Обычно она выполняется на магистралях сетей, которые передают очень большое количество индивидуальных потоков. Агрегирование потоков, имеющих общую часть маршрута через сеть, позволяет уменьшить количество хранимой промежуточными

узлами сети информации, так как агрегированные потоки описываются в них как одно целое. В результате снижается нагрузка на промежуточные узлы сети и повышается их быстродействие.

Признаки потока могут иметь *глобальное* или *локальное* значение — в первом случае они однозначно определяют поток в пределах всей сети, а во втором — в пределах одного транзитного узла. Пара идентифицирующих поток адресов конечных узлов — это пример глобального признака. Примером признака, локально определяющего поток в пределах устройства, может служить номер (идентификатор) интерфейса данного устройства, на который поступили данные. Например, возвращаясь к рис. 2.12, узел 1 может быть настроен так, чтобы передавать на интерфейс *B* все данные, поступившие с интерфейса *A*, а на интерфейс *C* — данные, поступившие с интерфейса *D*. Такое правило позволяет отделить поток данных узла 2 от потока данных узла 7 и направлять их для транзитной передачи через разные узлы сети, в данном случае поток узла 2 — через узел 5, а поток узла 7 — через узел 8.

Метка потока — это особый тип признака. Она представляет собой некоторое число, которое несут все данные потока. **Глобальная метка** назначается данным потока и не меняет своего значения на всем протяжении его пути следования от узла источника до узла назначения, таким образом, она уникально определяет поток в пределах сети. В некоторых технологиях используются **локальные метки** потока, динамически меняющие свое значение при передаче данных от одного узла к другому.

Таким образом, распознавание потоков во время коммутации происходит на основании признаков, в качестве которых помимо обязательного адреса назначения данных могут выступать и другие признаки, такие, например, как идентификаторы приложений.

Маршрутизация

Задача маршрутизации в свою очередь включает в себя две подзадачи:

- определение маршрута;
- оповещение сети о выбранном маршруте.

Определить маршрут означает выбрать последовательность транзитных узлов и их интерфейсов, через которые надо передавать данные, чтобы доставить их адресату. Определение маршрута — сложная задача, особенно когда конфигурация сети такова, что между парой взаимодействующих сетевых интерфейсов существует множество путей. Чаще всего выбор останавливают на одном *оптимальном*[1] по некоторому критерию маршруте. В качестве критериев оптимальности могут выступать, например, номинальная пропускная способность и загруженность каналов связи; задержки, вносимые каналами; количество промежуточных транзитных узлов; надежность каналов и транзитных узлов.

Но даже в том случае, когда между конечными узлами существует только *один* путь, при сложной топологии сети его нахождение может представлять собой нетривиальную задачу.

Маршрут может определяться эмпирически («вручную») администратором сети на основании различных, часто не формализуемых соображений. Среди побудительных мотивов вы-

[1] На практике для снижения объема вычислений ограничиваются поиском не оптимального в математическом смысле, а рационального, то есть близкого к оптимальному, маршрута.

бора пути могут быть: особые требования к сети со стороны различных типов приложений, решение передавать трафик через сеть определенного поставщика услуг, предположения о пиковых нагрузках на некоторые каналы сети, соображения безопасности.

Однако эмпирический подход к определению маршрутов мало пригоден для большой сети со сложной топологией. В этом случае используются автоматические методы определения маршрутов. Для этого конечные узлы и другие устройства сети оснащаются специальными программными средствами, которые организуют взаимный обмен служебными сообщениями, позволяющий каждому узлу составить свое «представление» о сети. Затем на основе собранных данных программными методами определяются рациональные маршруты.

При выборе маршрута часто ограничиваются только информацией о топологии сети. Этот подход иллюстрирует рис. 2.13. Для передачи трафика между конечными узлами *A* и *C* существуют два альтернативных маршрута: *A*-1-2-3-*C* и *A*-1-3-*C*. Если мы учитываем только топологию, то выбор очевиден — маршрут *A*-1-3-*C*, который имеет меньше транзитных узлов.

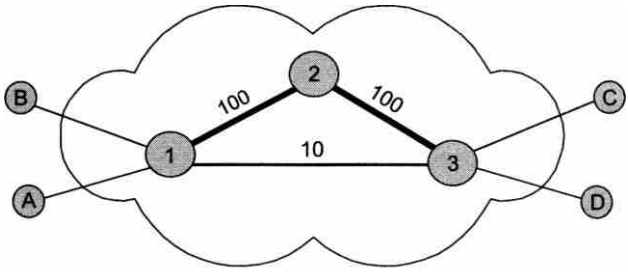


Рис. 2.13. Выбор маршрута

Решение было найдено путем минимизации критерия, в качестве которого в данном примере выступала длина маршрута, измеренная количеством транзитных узлов. Однако, возможно, наш выбор был не самым лучшим. На рисунке показано, что каналы 1-2 и 2-3 обладают пропускной способностью 100 Мбит/с, а канал 1-3 — только 10 Мбит/с. Если мы хотим, чтобы наша информация передавалась по сети с максимально возможной скоростью, то нам следовало бы выбрать маршрут *A*-1-2-3-*C*, хотя он и проходит через большее количество промежуточных узлов. То есть можно сказать, что маршрут *A*-1-2-3-*C* в данном случае оказывается «более коротким».

Абстрактная оценка условного «расстояния» между двумя узлами сети называется **метрикой**. Так, для измерения длины маршрута могут быть использованы разные метрики — количество транзитных узлов, как в предыдущем примере, линейная протяженность маршрута и даже его стоимость в денежном выражении. Для построения метрики, учитывающей пропускную способность, часто применяют следующий прием: длину каждого канала-участка характеризуют величиной, обратной его пропускной способности. Чтобы оперировать целыми числами, выбирают некоторую константу, заведомо большую, чем пропускные способности каналов в сети. Например, если мы в качестве такой константы выберем 100 Мбит/с, то метрика каждого из каналов 1-2 и 2-3 равна 1, а метрика канала 1-3 составляет 10. Метрика маршрута равна сумме метрик составляющих его каналов, поэтому часть пути 1-2-3 обладает метрикой 2, а альтернативная часть пути 1-3 — метрикой 10. Мы выбираем более «короткий» путь, то есть путь *A*-1-2-3-*C*.

Описанные подходы к выбору маршрутов не учитывают текущую степень загруженности каналов трафиком[1]. Используя аналогию с автомобильным трафиком, можно сказать, что мы выбирали маршрут по карте, учитывая количество промежуточных городов и ширину дороги (аналог пропускной способности канала), отдавая предпочтение скоростным магистралям. Но мы не стали слушать радио- или телепрограмму, которая сообщает о текущих заторах на дорогах. Так что наше решение оказывается отнюдь не лучшим, когда по маршруту *A*-1-2-3-*C* уже передается большое количество потоков, а маршрут *A*-1-3-*C* практически свободен.

После того как маршрут определен (вручную или автоматически), надо *оповестить* о нем все устройства сети. Сообщение о маршруте должно нести каждому транзитному устройству примерно такую информацию: «Каждый раз, когда в устройство поступят данные, относящиеся к потоку *n*, их следует передать для дальнейшего продвижения на интерфейс *if*1». Каждое подобное сообщение о маршруте обрабатывается транзитным устройством, в результате создается новая запись в **таблице коммутации** (называемой также **таблицей маршрутизации**). В этой таблице локальному или глобальному признаку (признакам) потока (например, метке, номеру входного интерфейса или адресу назначения) ставится в соответствие номер интерфейса, на который устройство должно передавать данные, относящиеся к этому потоку.

Таблица 2.1 является фрагментом таблицы коммутации, содержащий запись, сделанную на основании сообщения о необходимости передачи потока *n* на интерфейс *if*1.

Таблица 2.1. Фрагмент таблицы коммутации

Признаки потока	Направление передачи данных (номер интерфейса и/или адрес следующего узла)
n ={DA, SA, A}	if1

В этой таблице в качестве признака потока использованы адрес назначения DA, адрес источника SA и тип приложения A, который генерирует пакеты потока.

Конечно, детальное описание структуры сообщения о маршруте и содержимого таблицы коммутации зависит от конкретной технологии, однако эти особенности не меняют сущности рассматриваемых процессов. Чаще всего в качестве признака потока используется адрес назначения пакета.

Передача информации транзитным устройствам о выбранных маршрутах, так же как и определение маршрута, может осуществляться вручную или автоматически. Администратор сети может зафиксировать маршрут, выполнив в ручном режиме конфигурирование устройства, например жестко скоммутировав на длительное время определенные пары входных и выходных интерфейсов (как работали «телефонные барышни» на первых

[1] Такие методы, в которых используется информация о текущей загруженности каналов связи, позволяют определять более рациональные маршруты, однако требуют интенсивного обмена служебной информацией между узлами сети.

коммутаторах). Он может также по собственной инициативе внести запись о маршруте в таблицу коммутации.

Однако поскольку топология и состав информационных потоков могут меняться (отказы узлов или появление новых промежуточных узлов, изменение адресов или определение новых потоков), гибкое решение задач определения и задания маршрутов предполагает постоянный анализ состояния сети и обновление маршрутов и таблиц коммутации. В таких случаях задачи прокладки маршрутов, как правило, не могут быть решены без достаточно сложных программных и аппаратных средств.

Продвижение данных

Итак, пусть маршруты определены, записи о них сделаны в таблицах всех транзитных узлов, все готово к выполнению основной операции — передаче данных между абонентами (коммутации абонентов).

Для каждой пары абонентов эта операция может быть представлена несколькими (по числу транзитных узлов) *локальными* операциями коммутации. Прежде всего отправитель должен выставить данные на тот свой интерфейс, с которого начинается найденный маршрут, а все транзитные узлы должны соответствующим образом выполнить «переброску» данных с одного своего интерфейса на другой, другими словами, выполнить **коммутацию интерфейсов**. Устройство, функциональным назначением которого является коммутация, называется **коммутатором**. На рис. 2.14 показан коммутатор, который переключает информационные потоки между четырьмя своими интерфейсами.

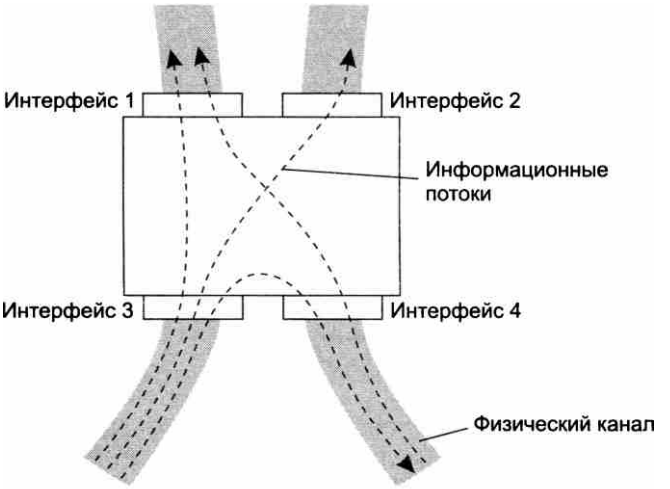


Рис. 2.14. Коммутатор

Прежде чем выполнить коммутацию, коммутатор должен распознать поток. Для этого в поступивших данных коммутатор пытается найти признак какого-либо из потоков, заданных в его таблице коммутации. Если произошло совпадение, то эти данные направляются на интерфейс, определенный для них в маршруте.

О ТЕРМИНАХ

Термины «коммутация», «таблица коммутации» и «коммутатор» в телекоммуникационных сетях могут трактоваться неоднозначно. Мы уже определили коммутацию как процесс соединения абонентов сети через транзитные узлы. Этим же термином мы обозначаем и соединение интерфейсов в пределах отдельного транзитного узла. Коммутатором в широком смысле называется устройство любого типа, способное выполнять операции переключения потока данных с одного интерфейса на другой. Операция коммутации может выполняться в соответствии с различными правилами и алгоритмами. Некоторые способы коммутации и соответствующие им таблицы и устройства получили специальные названия. Например, в технологии IP для обозначения аналогичных понятий используются термины «маршрутизация», «таблица маршрутизации», «маршрутизатор». В то же время за другими специальными типами коммутации и соответствующими устройствами закрепились те же самые названия «коммутация», «таблица коммутации» и «коммутатор», применяемые в узком смысле, например как коммутация и коммутатор в локальной сети Ethernet. Для телефонных сетей, которые появились намного раньше компьютерных, также характерна аналогичная терминология, «коммутатор» является здесь синонимом «телефонной станции».

Коммутатором может быть как специализированное устройство, так и универсальный компьютер со встроенным программным механизмом коммутации, в этом случае коммутатор называется программным. Компьютер может совмещать функции коммутации данных с выполнением своих обычных функций как конечного узла. Однако во многих случаях более рациональным является решение, в соответствии с которым некоторые узлы в сети выделяются *специально* для коммутации. Эти узлы образуют **коммутационную сеть**, к которой подключаются все остальные. На рис. 2.15 показана коммутационная сеть, образованная из узлов 1, 5, 6 и 8, к которой подключаются конечные узлы 2, 3, 4, 7, 9 и 10.

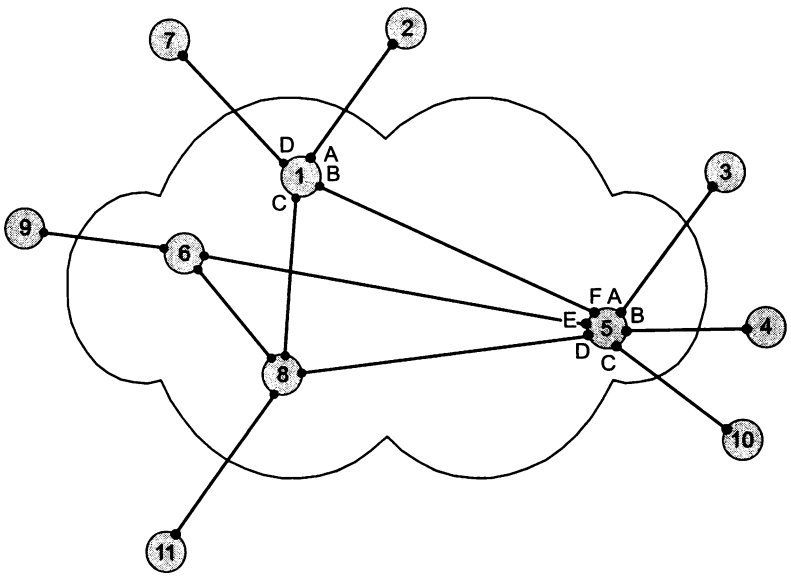


Рис. 2.15. Коммутационная сеть

Мультиплексирование и демультиплексирование

Чтобы определить, на какой интерфейс следует передать поступившие данные, коммутатор должен выяснить, к какому потоку они относятся. Эта задача должна решаться независимо от того, поступает на вход коммутатора только один «чистый» поток или «смешанный» поток, являющийся результатом агрегирования нескольких потоков. В последнем случае к задаче распознавания потоков добавляется задача демультиплексирования.

Демультиплексирование — разделение суммарного потока на несколько составляющих его потоков.

Как правило, операцию коммутации сопровождает также обратная операция мультиплексирования.

Мультиплексирование (агрегирование) — образование из нескольких отдельных потоков общего агрегированного потока, который передается по одному физическому каналу связи.

Другими словами, мультиплексирование — это способ разделения одного имеющегося физического канала между несколькими одновременно протекающими сеансами связи абонентов сети.

Операции мультиплексирования/демультиплексирования имеют такое же важное значение в любой сети, как и операции коммутации, потому что без них пришлось бы для каждого потока предусматривать отдельный канал, что привело бы к большому количеству параллельных связей в сети и свело бы на нет все преимущества неполносвязной сети.

На рис. 2.16 показан фрагмент сети, состоящий из трех коммутаторов. Коммутатор 1 имеет четыре сетевых интерфейса. На интерфейс 1 поступают данные с двух интерфейсов — 3 и 4. Их надо передать в общий физический канал, то есть выполнить операцию мультиплексирования.

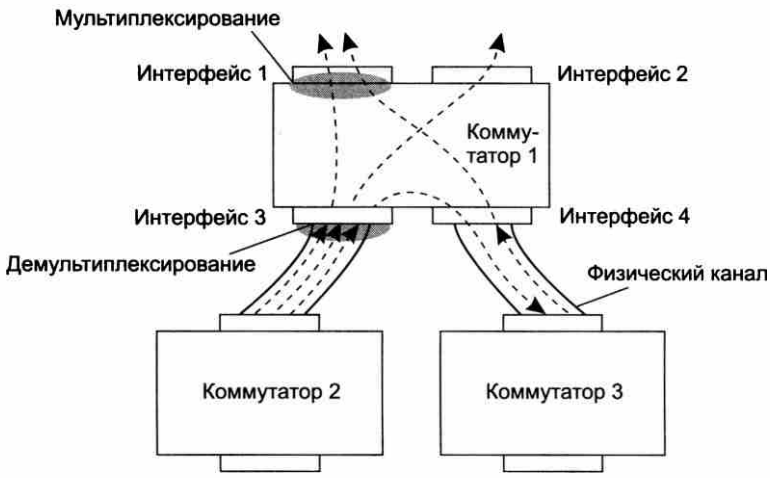


Рис. 2.16. Операции мультиплексирования и демультиплексирования потоков при коммутации

Одним из основных способов мультиплексирования потоков является **разделение времени**. При этом способе каждый поток время от времени (с фиксированным или случайным периодом) получает физический канал в полное свое распоряжение и передает по нему свои данные. Распространено также **частотное разделение** канала, когда каждый поток передает данные в выделенном ему частотном диапазоне.

Технология мультиплексирования должна позволять получателю такого суммарного потока выполнять обратную операцию — разделение (демультиплексирование) данных на слагаемые потоки. На интерфейсе 3 коммутатор выполняет демультиплексирование потока на три составляющих его подпотока. Один из них он передает на интерфейс 1, другой — на интерфейс 2, третий — на интерфейс 4.

Вообще говоря, на одном интерфейсе могут одновременно выполняться обе функции — мультиплексирование и демультиплексирование.

Частный случай коммутатора, у которого все входящие информационные потоки коммутируются на один выходной интерфейс, где они мультиплексируются в один агрегированный поток, называется **мультиплексором**. Коммутатор, который имеет один входной интерфейс и несколько выходных, называется **демультиплексором** (рис. 2.17).

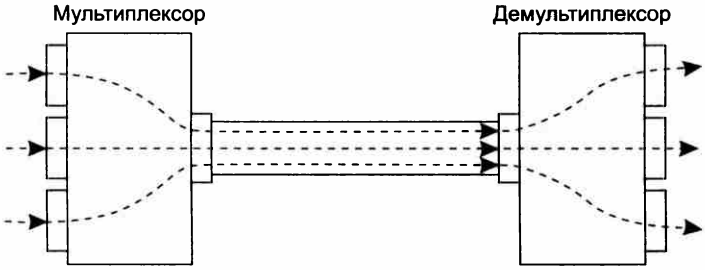


Рис. 2.17. Мультиплексор и демультиплексор

Разделяемая среда передачи данных

Во всех рассмотренных ранее примерах мультиплексирования потоков к каждой линии связи подключались только два интерфейса. В том случае, когда линия связи является дуплексным каналом связи, как это показано на рис. 2.18, каждый из интерфейсов монопольно использует канал связи в направлении «от себя». Это объясняется тем, что дуплексный канал состоит из двух независимых сред передачи данных (подканалов), и так как только передатчик интерфейса является активным устройством, а приемник пассивно ожидает поступления сигналов от приемника, то конкуренции поканалов не возникает. Такой режим использования среды передачи данных является в настоящее время основным в компьютерных локальных и глобальных сетях.

Однако если в глобальных сетях такой режим использовался всегда, то в локальных сетях до середины 90-х годов преобладал другой режим, основанный на разделяемой среде передачи данных.

В наиболее простом случае эффект разделения среды возникает при соединении двух интерфейсов с помощью полудуплексного канала связи, то есть такого канала, который

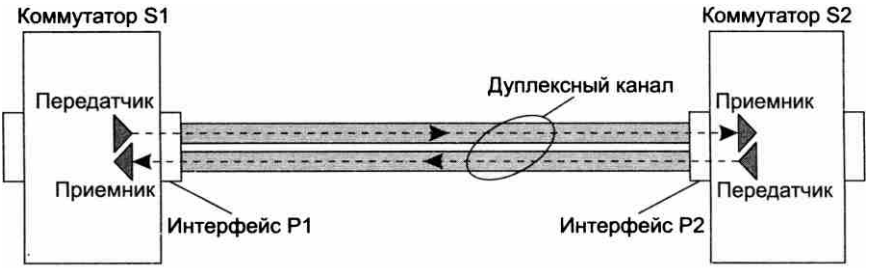


Рис. 2.18. Дуплексный канал — разделяемая среда отсутствует

Разделяемой средой (shared medium) называется физическая среда передачи данных, к которой непосредственно подключено несколько передатчиков узлов сети. Причем в каждый момент времени только один из передатчиков какого-либо узла сети получает доступ к разделяемой среде и использует ее для передачи данных приемнику другого узла, подключенному к этой же среде.

может передавать данные в любом направлении, но только попеременно (рис. 2.19). В этом случае к одной и той же среде передачи данных (например, к коаксиальному кабелю или общей радиосреде) подключены два приемника двух независимых узлов сети.

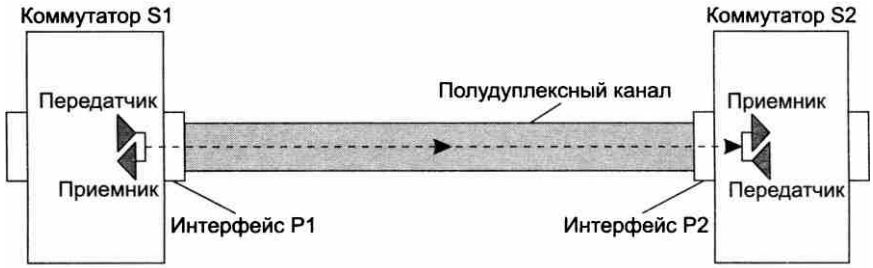


Рис. 2.19. Полудуплексный канал — разделяемая среда

При таком применении среды передачи данных возникает новая задача *совместного использования среды независимыми передатчиками* таким образом, чтобы в каждый отдельный момент времени по среде передавались данные только одного передатчика. Другими словами, возникает *необходимость в механизме синхронизации доступа интерфейсов к разделяемой среде.*

Обобщением разделяемой среды является случай, показанный на рис. 2.20, когда к каналу связи подключаются более двух интерфейсов (в приведенном примере — три), при этом применяется топология общей шины.

Существуют различные способы решения задачи организации совместного доступа к разделяемым линиям связи. Одни из них подразумевают *централизованный* подход, когда доступом к каналу управляет специальное устройство — **арбитр**, другие — *децентрализованный.* Если мы обратимся к организации работы компьютера, то увидим, что доступ к системной шине компьютера, которую совместно используют внутренние блоки компьютера, управляется централизованно — либо процессором, либо специальным арбитром шины.

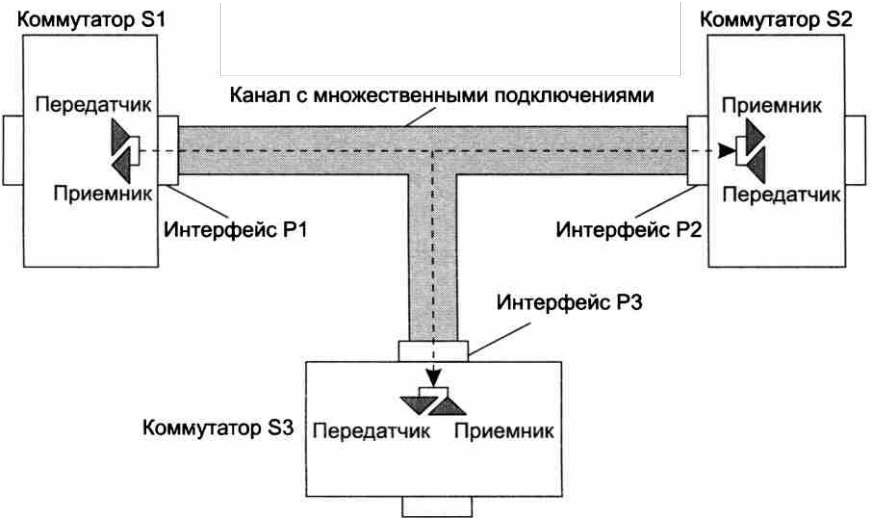


Рис. 2.20. Канал с множественными подключениями — разделяемая среда

В сетях организация совместного доступа к линиям связи имеет свою специфику из-за существенно большего времени распространения сигналов по линиям связи. Здесь процедуры согласования доступа к линии связи могут занимать слишком большой промежуток времени и приводить к значительным потерям производительности сети. Именно по этой причине механизм разделения среды в глобальных сетях практически не используется.

На первый взгляд может показаться, что механизм разделения среды очень похож на механизм мультиплексирования потоков — в том и другом случае по линии связи передается несколько потоков данных. Однако здесь есть принципиальное различие, касающееся того, как контролируется (управляется) линия связи. При мультиплексировании дуплексная линия связи в каждом направлении находится под полным контролем одного коммутатора, который решает, какие потоки разделяют общий канал связи.

Для локальных сетей разделяемая среда сравнительно долго была основным механизмом использования каналов связи, который применялся во всех технологиях локальных сетей — Ethernet, Token Ring, FDDI. При этом в технологиях локальных сетей применялись децентрализованные методы доступа к среде, не требующие наличия арбитра в сети. Популярность техники разделения среды в локальных сетях объяснялась простотой и экономичностью аппаратных решений. Например, для создания сети Ethernet на коаксиальном кабеле никакого другого сетевого оборудования, кроме сетевых адаптеров компьютеров и самого кабеля, не требуется. Наращивание количества компьютеров в локальной сети Ethernet на коаксиальном кабеле выполняется также достаточно просто — путем присоединения нового отрезка кабеля к существующему.

Сегодня в проводных локальных сетях метод разделения среды практически перестал применяться. Основной причиной отказа от разделяемой среды явилась ее низкая и плохо предсказуемая производительность, а также плохая масштабируемость[1]. Низкая про-

[1] Масштабируемостью называют свойство сети допускать наращивание количества узлов и протяженность линий связи в очень широких пределах без снижения производительности.

изводительность объясняется тем, что пропускная способность канала связи делится между всеми компьютерами сети. Например, если локальная сеть Ethernet состоит из 100 компьютеров, а для их связи используются коаксиальный кабель и сетевые адаптеры, работающие на скорости 10 Мбит/с, то в среднем на каждый компьютер приходится только 0,1 Мбит/с пропускной способности. Более точно оценить долю пропускной способности, приходящуюся на какой-либо компьютер сети, трудно, так как эта величина зависит от многих случайных факторов, например активности других компьютеров. Наверное, к этому моменту читателю уже понятна причина плохой масштабируемости подобной сети — чем больше мы добавляем компьютеров, тем меньшая доля пропускной способности достается каждому компьютеру сети.

Описанные недостатки являются следствием самого принципа разделения среды, поэтому преодолеть их полностью невозможно. Появление в начале 90-х недорогих коммутаторов локальных сетей привело к настоящей революции в этой области, и постепенно коммутаторы вытеснили разделяемую среду полностью.

Сегодня механизм разделения среды используется только в беспроводных локальных сетях, где среда — радиоэфир — естественным образом соединяет все конечные узлы, находящиеся в зоне распространения сигнала.

Типы коммутации

Комплекс технических решений обобщенной задачи коммутации в своей совокупности составляет основу любой сетевой технологии. Как уже отмечалось, к этим частным задачам относятся:

- определение потоков и соответствующих маршрутов;
- фиксация маршрутов в конфигурационных параметрах и таблицах сетевых устройств;
- распознавание потоков и передача данных между интерфейсами одного устройства;
- мультиплексирование/демультиплексирование потоков;
- разделение среды передачи.

Среди множества возможных подходов к решению задачи коммутации абонентов в сетях выделяют два основополагающих, к которым относят **коммутацию каналов** и **коммутацию пакетов**.

Каждый из этих двух подходов имеет свои достоинства и недостатки. Существуют традиционные области применения каждой из техник коммутации, например телефонные сети строились и продолжают строиться с использованием техники коммутации каналов, а компьютерные сети в подавляющем большинстве основаны на технике коммутации пакетов. Техника коммутации пакетов гораздо моложе своей конкурентки и пытается вытеснить ее из некоторых областей, например из телефонии (в форме интернет- или IP-телефонии), но этот спор пока не решен, и скорее всего, две техники коммутации будут сосуществовать еще долгое время, дополняя друг друга. Тем не менее по долгосрочным прогнозам многих специалистов будущее принадлежит технике коммутации пакетов как более гибкой и универсальной.

ПРИМЕР-АНАЛОГИЯ

Поясним достаточно абстрактное описание обобщенной модели коммутации на примере работы традиционной почтовой службы. Почта тоже работает с информационными потоками, которые в данном случае составляют почтовые отправления. Основным признаком почтового потока является адрес получателя. Для упрощения будем рассматривать в качестве адреса только страну, например Индия, Норвегия, Россия, Бразилия и т. д. Дополнительным признаком потока может служить особое требование к надежности или скорости доставки. Например, пометка «Avia» на почтовых отправлениях в Бразилию выделит из общего потока почты в Бразилию подпоток, который будет доставляться самолетом.

Для каждого потока почтовая служба должна определить маршрут, который будет проходить через последовательность почтовых отделений, являющихся аналогами коммутаторов. В результате многолетней работы почтовой службы уже определены маршруты для большинства адресов назначения. Иногда возникают новые маршруты, связанные с появлением новых возможностей — политических, транспортных, экономических. После выбора нового маршрута нужно оповестить о нем сеть почтовых отделений. Как видно, эти действия очень напоминают работу телекоммуникационной сети. Информация о выбранных маршрутах следования почты представлена в каждом почтовом отделении в виде таблицы, в которой задано соответствие между страной назначения и следующим почтовым отделением. Например, в почтовом отделении города Саратова все письма, адресованные в Индию, направляются в почтовое отделение Ашхабада, а письма, адресованные в Норвегию, — в почтовое отделение Санкт-Петербурга. Такая таблица направлений доставки почты является прямой аналогией таблицы коммутации коммуникационной сети.

Каждое почтовое отделение работает подобно коммутатору. Все поступающие от абонентов и других почтовых отделений почтовые отправления сортируются, то есть происходит распознавание потоков. После этого почтовые отправления, принадлежащие одному «потоку», упаковываются в мешок, для которого в соответствии с таблицей направлений определяется следующее по маршруту почтовое отделение.

Выводы

Для того чтобы пользователь сети получил возможность доступа к ресурсам «чужих» компьютеров, таким как диски, принтеры, плоттеры, необходимо дополнить все компьютеры сети специальными средствами. В каждом компьютере функции передачи данных в линию связи совместно выполняют аппаратный модуль, называемый сетевым адаптером, или сетевой интерфейсной картой, и управляющая программа — драйвер. Задачи более высокого уровня — формирование запросов к ресурсам и их выполнение — решают соответственно клиентские и серверные модули ОС.

Даже в простейшей сети, состоящей из двух компьютеров, возникают проблемы физической передачи сигналов по линиям связи: кодирование и модуляция, синхронизация передающего и принимающего устройств, контроль корректности переданных данных.

Важными характеристиками, связанными с передачей трафика через физические каналы, являются: предложенная нагрузка, скорость передачи данных, пропускная способность, емкость канала связи, полоса пропускания.

При связывании в сеть более двух компьютеров возникают проблемы выбора топологии (полносвязной, звезды, кольца, общей шины, иерархического дерева, произвольной); способа адресации (плоского или иерархического, числового или символьного); способа разделения линий связи и механизма коммутации.

В неполносвязных сетях соединение пользователей осуществляется путем коммутации через сеть транзитных узлов. При этом должны быть решены следующие задачи: определение потоков данных и маршрутов для них, продвижение данных в каждом транзитном узле, мультиплексирование и демультиплексирование потоков.

Среди множества возможных подходов к решению задачи коммутации выделяют два основополагающих — коммутацию каналов и пакетов.

Контрольные вопросы

1. Какие из перечисленных терминов в некотором контексте могут использоваться как синонимы? Варианты ответов:
   а) емкость канала связи;
   б) скорость передачи данных;
   в) полоса пропускания канала связи;
   г) пропускная способность канала связи.
2. Каким типом адреса снабжают посылаемые данные, когда хотят, чтобы они были доставлены всем узлам сети? Варианты ответов:
   а) multicast;
   б) anycast;
   в) broadcast;
   г) unicast.
3. Какие признаки могут быть использованы для определения информационного потока? Варианты ответов:
   а) адрес назначения;
   б) адрес источника;
   в) тип приложения;
   г) номер интерфейса, на который поступил пакет;
   д) все перечисленные.
4. Что можно считать недостатком метода нахождения маршрута по критерию минимума промежуточных узлов? Варианты ответов:
   а) не учитывается пропускная способность линий связи;
   б) не учитывается загрузки линий связи;
   в) не учитывается топология сети.
5. Могут ли клиентская и серверная части приложения работать на одном и том же компьютере?

ГЛАВА 3 Коммутация каналов и пакетов

Коммутация каналов

Исторически коммутация каналов появилась намного раньше коммутации пакетов и ведет свое происхождение от первых телефонных сетей. Невозможность динамического перераспределения пропускной способности физического канала является принципиальным ограничением сети с коммутацией каналов.

Принцип коммутации пакетов был предложен разработчиками компьютерных сетей. При коммутации пакетов учитываются особенности компьютерного трафика, поэтому данный способ коммутации является более эффективным для компьютерных сетей по сравнению с традиционным методом коммутации каналов, применяющимся в телефонных сетях. Однако достоинства и недостатки любой сетевой технологии относительны. Наличие буферной памяти в коммутаторах сетей с коммутацией пакетов позволяет эффективно использовать пропускную способность каналов при передаче пульсирующего трафика, но приводит к случайным задержкам в доставке пакетов, что для трафика реального времени является серьезным недостатком.

Сети, построенные на принципе коммутации каналов, имеют богатую историю, они и сегодня находят широкое применение в мире телекоммуникаций, являясь основой высокоскоростных магистральных каналов связи. Первые сеансы связи между компьютерами были осуществлены через телефонную сеть, то есть также с применением техники коммутации каналов, а пользователи, которые получают доступ в Интернет по модему, продолжают обслуживаться этими сетями, так как их данные доходят до оборудования провайдера по местной телефонной сети.

В сетях с коммутацией каналов решаются все те частные задачи коммутации, которые были сформулированы ранее. Так, в качестве информационных потоков в сетях с коммутацией каналов выступают данные, которыми обмениваются пары **абонентов**. Соответственно глобальным признаком потока является пара адресов (телефонных номеров) абонентов, связывающихся между собой. Для всех возможных потоков заранее определяются маршруты. Маршруты в сетях с коммутацией каналов либо задаются «вручную» администратором сети, либо находятся автоматически с привлечением специальных программных и аппаратных средств. Маршруты фиксируются в таблицах, в которых признакам потока ставятся в соответствие идентификаторы выходных интерфейсов коммутаторов. На основании этих таблиц происходят продвижение и мультиплексирование данных. Однако, как уже было сказано, в сетях с коммутацией каналов решение всех этих задач имеет свои особенности.

Элементарный канал

Одной из особенностей сетей с коммутацией каналов является понятие элементарного канала.

Элементарный канал (или просто **канал**) — это базовая техническая характеристика сети с коммутацией каналов, представляющая собой некоторое фиксированное в пределах данного типа сетей значение пропускной способности. Любая линия связи в сети с коммутацией каналов имеет пропускную способность, кратную элементарному каналу, принятому для данного типа сети.

В предыдущих разделах мы использовали термин «канал» как синоним термина «линия связи». Говоря же о сетях с коммутацией каналов, мы придаем термину «канал» значение единицы пропускной способности.

Значение элементарного канала, или, другими словами, минимальная единица пропускной способности линии связи, выбирается с учетом разных факторов. Очевидно, однако, что элементарный канал не стоит выбирать меньше минимально необходимой пропускной способности для передачи ожидаемой нагрузки. Например, в традиционных телефонных сетях наиболее распространенным значением элементарного канала сегодня является скорость 64 Кбит/с — это минимально достаточная скорость для качественной цифровой передачи голоса.

ОЦИФРОВЫВАНИЕ ГОЛОСА

Задача оцифровывания голоса является частным случаем более общей проблемы — передачи аналоговой информации в дискретной форме. Она была решена в 60-е годы, когда голос начал передаваться по телефонным сетям в виде последовательности единиц и нулей. Такое преобразование основано на дискретизации непрерывных процессов как по амплитуде, так и по времени (рис. 3.1).

Рис. 3.1. Дискретная модуляция непрерывного процесса

Амплитуда исходной непрерывной функции измеряется с заданным периодом — за счет этого происходит *дискретизация по времени*. Затем каждый замер представляется в виде двоичного числа определенной разрядности, что означает *дискретизацию по значениям* — непрерывное множество возможных значений амплитуды заменяется дискретным множеством ее значений.

Для качественной передачи голоса используется частота квантования амплитуды звуковых колебаний в 8000 Гц (дискретизация по времени с интервалом 125 мкс). Для представления амплитуды одного замера чаще всего используется 8 бит кода, что дает 256 градаций звукового сигнала (дискретизация по значениям). В этом случае для передачи одного голосового канала необходима пропускная способность 64 Кбит/с: 8000 × 8 = 64 000 бит/с, или 64 Кбит/с. Такой голосовой канал называют **элементарным каналом цифровых телефонных сетей**.

Линии связи в сетях с коммутацией пакетов (как, впрочем, и в остальных типах компьютерных сетей) имеют разную пропускную способность, одни — большую, другие — меньшую. Выбирая линии связи с разными скоростными качествами, специалисты, проектирующие сеть, стараются учесть разную интенсивность информационных потоков, которые могут возникнуть в разных фрагментах сети — чем ближе к центру сети, тем выше пропускная способность линии связи, так как магистральные линии агрегируют трафик большого количества периферийных линий связи.

Особенностью сетей с коммутацией каналов является то, что пропускная способность каждой линии связи должна быть равна *целому числу* элементарных каналов.

Так, линии связи, подключающие абонентов к телефонной сети, могут содержать 2, 24 или 30 элементарных каналов, а линии, соединяющие коммутаторы, — 480 или 1920 каналов.

Обратимся к фрагменту сети, изображенному на рис. 3.2. Предположим, что эта сеть характеризуется элементарным каналом *P* бит/с. В сети существуют линии связи разной

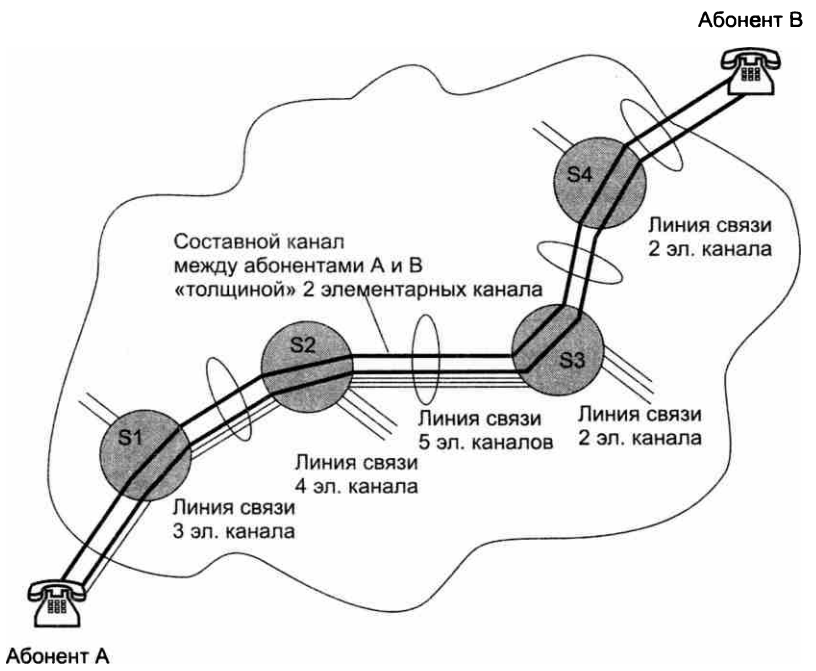


Рис. 3.2. Составной канал в сети с коммутацией каналов

пропускной способности, состоящие из 2, 3, 4 и 5 элементарных каналов. На рисунке показаны два абонента, *A* и *B*, генерирующие во время сеанса связи (телефонного разговора) *информационный поток*, для которого в сети был предусмотрен *маршрут*, проходящий через четыре коммутатора — *S*1, *S*2, *S*3 и *S*4. Предположим также, что интенсивность информационного потока между абонентами не превосходит 2*P* бит/с. Тогда для обмена данными этим двум абонентам достаточно иметь в своем распоряжении по паре элементарных каналов, «выделенных» из каждой линии связи, лежащей на маршруте следования данных от пункта *A* к пункту *B*. На рисунке эти элементарные каналы, необходимые абонентам *A* и *B*, обозначены толстыми линиями.

Составной канал

Канал, построенный путем коммутации (соединения) элементарных каналов, называют **составным каналом**.

В рассматриваемом примере для соединения абонентов *A* и *B* был создан составной канал «толщиной» в два элементарных канала. Если изменить наше предположение и считать, что предложенная нагрузка гарантированно не превысит *P* бит/с, то абонентам будет достаточно иметь в своем распоряжении составной канал «толщиной» в один элементарный канал. В то же время абоненты, интенсивно обменивающиеся данными, могут предъявить и более высокие требования к пропускной способности составного канала. Для этого они должны в каждой линии связи зарезервировать за собой большее (но непременно одинаковое для всех линий связи) количество элементарных каналов.

Подчеркнем следующие свойства составного канала:

- составной канал на всем своем протяжении состоит из *одинакового* количества элементарных каналов;
- составной канал имеет *постоянную и фиксированную пропускную способность* на всем своем протяжении;
- составной канал создается *временно* на период сеанса связи двух абонентов;
- на время сеанса связи все элементарные каналы, входящие в составной канал, поступают в *исключительное* пользование абонентов, для которых был создан этот составной канал;
- в течение всего сеанса связи абоненты могут посылать в сеть данные со скоростью, не превышающей пропускную способность составного канала;
- данные, поступившие в составной канал, гарантированно доставляются вызываемому абоненту *без задержек, потерь и с той же скоростью* (скоростью источника) вне зависимости от того, существуют ли в это время в сети другие соединения или нет;
- после окончания сеанса связи элементарные каналы, входившие в соответствующий составной канал, *объявляются свободными* и возвращаются в пул распределяемых ресурсов для использования другими абонентами.

В сети может одновременно происходить несколько сеансов связи (обычная ситуация для телефонной сети, в которой одновременно передаются разговоры сотен и тысяч абонен-

тов). Разделение сети между сеансами связи происходит на уровне элементарных каналов. Например (см. рис. 3.2), мы можем предположить, что после того, как в линии связи *S*2-*S*3 было выделено два канала для связи абонентов *A* и *B*, оставшиеся три элементарных канала были распределены между тремя другими сеансами связи, проходившими в это же время и через эту же линию связи. Такое *мультиплексирование* позволяет одновременно передавать через каждый физический канал трафик нескольких логических соединений.

Мультиплексирование означает, что абоненты вынуждены конкурировать за ресурсы, в данном случае за элементарные каналы. Возможны ситуации, когда некоторая промежуточная линия связи уже исчерпала свободные элементарные каналы, тогда новый сеанс связи, маршрут которого пролегает через данную линию связи, не может состояться.

Для того чтобы распознать такие ситуации, обмен данными в сети с коммутацией каналов предваряется **процедурой установления соединения**. В соответствии с этой процедурой абонент, являющийся инициатором сеанса связи (например, абонент *A* в нашей сети), посылает в коммутационную сеть **запрос**, представляющий собой сообщение, в котором содержится адрес вызываемого абонента, например абонента *B*[1].

Цель запроса — проверить, можно ли образовать составной канал между вызывающим и вызываемым абонентами. А для этого требуется соблюдение двух условий: наличие требуемого числа свободных элементарных каналов в каждой линии связи, лежащей на пути от *A* к *B*, и незанятость вызываемого абонента в другом соединении.

Запрос перемещается по *маршруту*, определенному для информационного потока данной пары абонентов. При этом используются глобальные таблицы коммутации, ставящие в соответствие *глобальному* признаку потока (адресу вызываемого абонента) идентификатор выходного интерфейса коммутатора (как уже упоминалось, такие таблицы часто называют также таблицами маршрутизации).

Если в результате прохождения запроса от абонента *A* к абоненту *B* выяснилось, что ничто не препятствует установлению соединения, происходит *фиксация* составного канала. Для этого во всех коммутаторах вдоль пути от *A* до *B* создаются записи в *локальных таблицах коммутации*, в которых указывается соответствие между *локальными признаками потока* — номерами элементарных каналов, зарезервированных для этого сеанса связи. Только после этого составной канал считается установленным и абоненты *A* и *B* могут начать свой сеанс связи.

Таким образом, продвижение данных в сетях с коммутацией каналов происходит в два этапа.

1. В сеть поступает служебное сообщение — запрос, который несет адрес вызываемого абонента и инициирует создание составного канала.
2. По подготовленному составному каналу передается основной поток данных, для передачи которого уже не требуется никакой вспомогательной информации, в том числе адреса вызываемого абонента. Коммутация данных в коммутаторах выполняется на основе локальных признаков — номеров элементарных каналов.

Запросы на установление соединения не всегда завершаются успешно. Если на пути между вызывающим и вызываемым абонентами отсутствуют свободные элементарные каналы или вызываемый узел занят, то происходит **отказ в установлении соединения**. Например,

---

[1] В телефонной сети посылке запроса соответствует набор телефонного номера.

если во время сеанса связи абонентов *A* и *B* абонент *C* пошлет запрос в сеть на установление соединения с абонентом *D*, то он получит отказ, потому что оба необходимых ему элементарных канала, составляющих линию связи коммутаторов *S*3 и *S*4, уже выделены соединению абонентов *A* и *B* (рис. 3.3). При отказе в установлении соединения сеть информирует вызывающего абонента специальным сообщением[1]. Чем больше нагрузка на сеть, то есть чем больше соединений она в данный момент поддерживает, тем больше вероятность отказа в удовлетворении запроса на установление нового соединения.

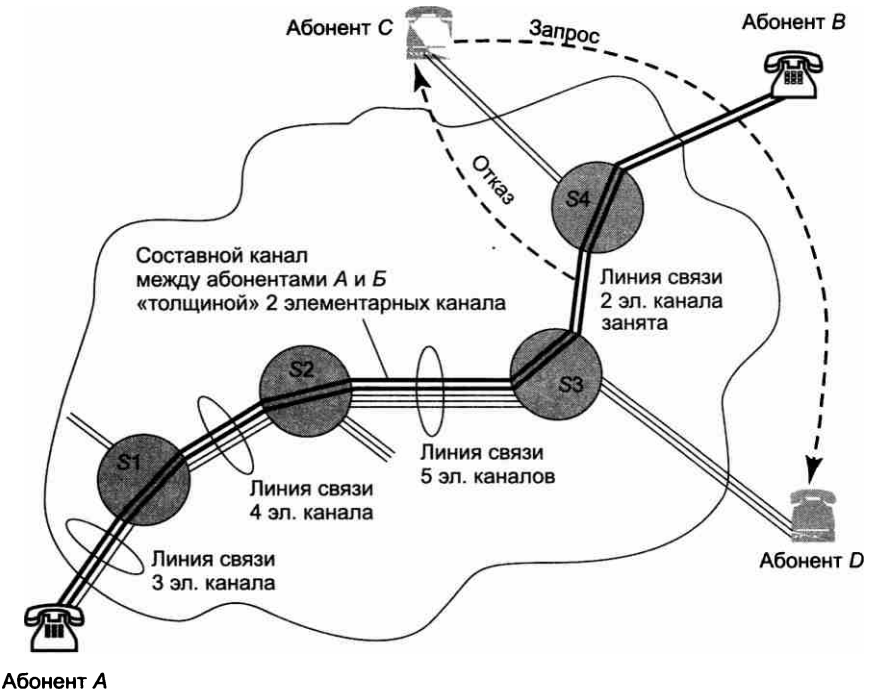


Рис. 3.3. Отказ в установлении соединения в сети с коммутацией каналов

Мы описали процедуру установления соединения в *автоматическом динамическом режиме*, основанном на способности абонентов отправлять в сеть служебные сообщения — запросы на установление соединения — и способности узлов сети обрабатывать такие сообщения. Подобный режим используется телефонными сетями: телефонный аппарат генерирует запрос, посылая в сеть импульсы (или тоновые сигналы), кодирующие номер вызываемого абонента, а сеть либо устанавливает соединение, либо сообщает об отказе сигналами «занято».

Однако это не единственно возможный режим работы сети с коммутацией каналов, существует и другой *статический ручной режим* установления соединения. Этот режим характерен для случаев, когда необходимо установить составной канал не на время одного сеанса

[1] Телефонная сеть в этом случае передает короткие гудки — сигнал «занято». Некоторые телефонные сети различают события «сеть занята» и «абонент занят», передавая гудки с разной частотой или используя разные тона.

связи абонентов, а на более долгий срок. Создание такого долговременного канала не могут инициировать абоненты, он создается администратором сети. Очевидно, что статический ручной режим мало пригоден для традиционной телефонной сети с ее короткими сеансами связи, однако он вполне оправдан для создания высокоскоростных телекоммуникационных каналов между городами и странами на более или менее постоянной основе.

Технология коммутации каналов ориентирована на минимизацию случайных событий в сети, то есть это технология, стремящаяся к детерминизму. Во избежание всяких возможных неопределенностей значительная часть работы по организации информационного обмена выполняется заранее, еще до того, как начнется собственно передача данных. Сначала по заданному адресу проверяется доступность необходимых элементарных каналов на всем пути от отправителя до адресата. Затем эти каналы закрепляются на все время сеанса для исключительного использования двумя абонентами и коммутируются в один непрерывный «трубопровод» (составной канал), имеющий «шлюзовые задвижки» на стороне каждого из абонентов. После этой исчерпывающей подготовительной работы остается сделать самое малое: «открыть шлюзы» и позволить информационному потоку свободно и без помех «перетекать» между заданными точками сети (рис. 3.4).

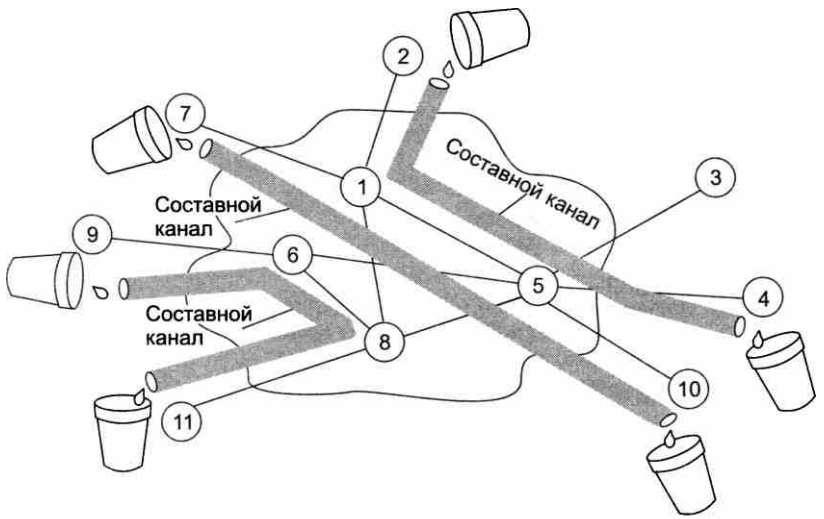


Рис. 3.4. Сеть с коммутацией каналов как система трубопроводов

Неэффективность передачи пульсирующего трафика

Сети с коммутацией каналов наиболее эффективно передают пользовательский трафик в том случае, когда скорость его постоянна в течение всего сеанса связи и максимально соответствует *фиксированной* пропускной способности физических линий связи сети. Эффективность работы сети снижается, когда информационные потоки, генерируемые абонентами, приобретают *пульсирующий* характер.

Так, разговаривая по телефону, люди постоянно меняют темп речи, перемежая быстрые высказывания паузами. В результате соответствующие «голосовые» информационные по-

токи становятся неравномерными, а значит, снижается эффективность передачи данных. Правда, в случае телефонных разговоров это снижение оказывается вполне приемлемым и позволяет широко использовать сети с коммутацией каналов для передачи голосового трафика.

Гораздо сильнее снижает эффективность сети с коммутацией каналов передача так называемого *компьютерного трафика*, то есть трафика, генерируемого приложениями, с которыми работает пользователь компьютера. Этот трафик практически всегда является пульсирующим. Например, когда вы загружаете из Интернета очередную страницу, скорость трафика резко возрастает, а после окончания загрузки падает практически до нуля. Если для описанного сеанса доступа в Интернет вы задействуете сеть с коммутацией каналов, то большую часть времени составной канал между вашим компьютером и веб-сервером будет простаивать. В то же время часть производительности сети окажется закрепленной за вами и останется недоступной другим пользователям сети. Сеть в такие периоды похожа на пустой эскалатор метро, который движется, но полезную работу не выполняет, другими словами, «перевозит воздух».

Для эффективной передачи неравномерного компьютерного трафика была специально разработана техника коммутации пакетов.

Коммутация пакетов

Сети с коммутацией пакетов, так же как и сети с коммутацией каналов, состоят из коммутаторов, связанных физическими линиями связи. Однако передача данных в этих сетях происходит совершенно по-другому. Образно говоря, по сравнению с сетью с коммутацией каналов сеть с коммутацией пакетов ведет себя менее «ответственно». Например, она может принять данные для передачи, не заботясь о резервировании линий связи на пути следования этих данных и не гарантируя требуемую пропускную способность. Сеть с коммутацией пакетов не создает заранее для своих абонентов отдельных каналов связи, выделенных исключительно для них. Данные могут задерживаться и даже теряться по пути следования. Как же при таком хаосе и неопределенности сеть с коммутацией пакетов выполняет свои функции по передаче данных?

Важнейшим принципом функционирования сетей с коммутацией пакетов является представление информации, передаваемой по сети, в виде структурно отделенных друг от друга порций данных, называемых **пакетами**[1].

Каждый пакет снабжен **заголовком** (рис. 3.5), в котором содержатся адрес назначения и другая вспомогательная информация (длина поля данных, контрольная сумма и др.), используемая для доставки пакета адресату. Наличие адреса в каждом пакете является одной из важнейших особенностей техники коммутации пакетов, так как каждый пакет может быть обработан коммутатором *независимо*[2] от других пакетов, составляющих сетевой

[1] Наряду с термином «пакет» используются также термины «кадр», «фрейм», «ячейка» и другие. В данном контексте различия в значении этих терминов несущественны.

[2] В некоторых технологиях коммутации пакетов (например, в технологии виртуальных каналов) полная независимость обработки пакетов не обеспечивается.

трафик. Помимо заголовка у пакета может иметься еще одно дополнительное поле, размещаемое в конце пакета и поэтому называемое **концевиком**. В концевике обычно помещается **контрольная сумма**, которая позволяет проверить, была ли искажена информация при передаче через сеть или нет.

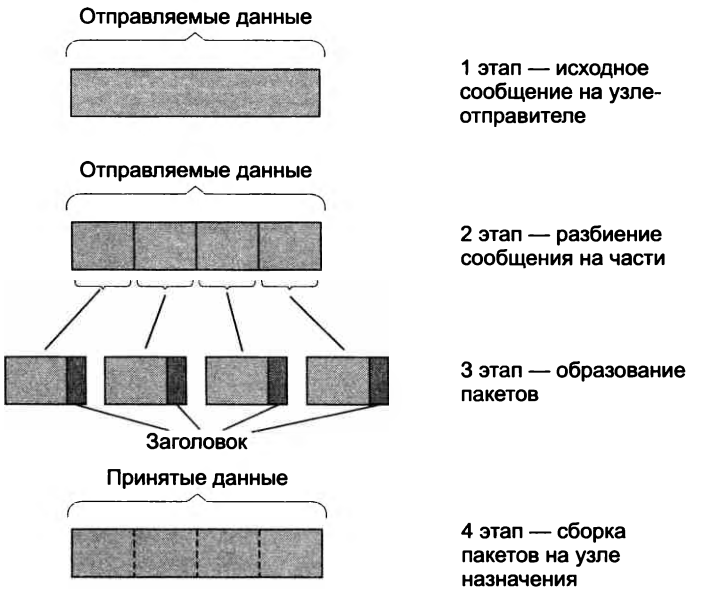


Рис. 3.5. Разбиение данных на пакеты

В зависимости от конкретной реализации технологии коммутации пакетов пакеты могут иметь фиксированную или переменную длину, кроме того, может меняться состав информации, размещенной в заголовках пакетов. Например, в технологии ATM пакеты (называемые там ячейками) имеют фиксированную длину, а в технологии Ethernet установлены лишь минимально и максимально возможные размеры пакетов (кадров).

Пакеты поступают в сеть *без предварительного резервирования линий связи* и *не с фиксированной заранее заданной скоростью*, как это делается в сетях с коммутацией каналов, а в том темпе, в котором их генерирует источник. Предполагается, что сеть с коммутацией пакетов в отличие от сети с коммутацией каналов всегда готова принять пакет от конечного узла.

ПРИМЕЧАНИЕ

Процедура резервирования пропускной способности может применяться и в сетях с коммутацией пакетов. Однако основная идея такого резервирования принципиально отличается от идеи резервирования пропускной способности в сетях с коммутацией каналов. Разница заключается в том, что пропускная способность канала сети с коммутацией пакетов может динамически перераспределяться между информационными потоками в зависимости от текущих потребностей каждого потока, чего не может обеспечить техника коммутации каналов. С деталями такого резервирования вы познакомитесь позже, в главе 6.

Как и в сетях с коммутацией каналов, в сетях с коммутацией пакетов для каждого из потоков вручную или автоматически определяется маршрут, фиксируемый в хранящихся на коммутаторах таблицах коммутации. Пакеты, попадая на коммутатор, обрабатываются и направляются по тому или иному маршруту на основании информации, содержащейся в их заголовках, а также в таблице коммутации (рис. 3.6).

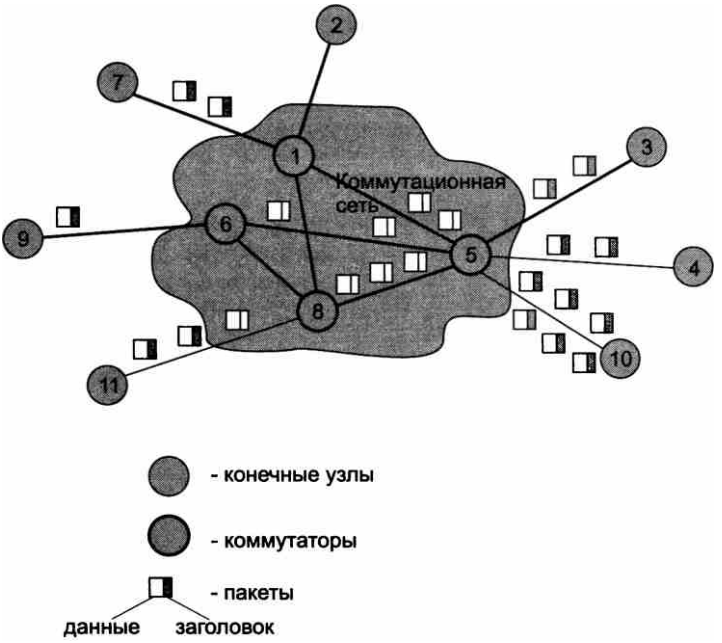


Рис. 3.6. Передача данных по сети в виде пакетов

Пакеты, принадлежащие как одному и тому же, так и разным информационным потокам, при перемещении по сети могут «перемешиваться» между собой, образовывать очереди и «тормозить» друг друга. На пути пакетов могут встречаться линии связи, имеющие разную пропускную способность. В зависимости от времени суток может сильно меняться и степень загруженности линий связи. В таких условиях не исключены ситуации, когда пакеты, принадлежащие одному и тому же потоку, могут перемещаться по сети с разными скоростями и даже прийти к месту назначения не в том порядке, в котором они были отправлены.

Разделение данных на пакеты позволяет передавать неравномерный компьютерный трафик более эффективно, чем в сетях с коммутацией каналов. Это объясняется тем, что пульсации трафика от отдельных компьютеров носят случайный характер и распределяются во времени так, что их пики чаще всего не совпадают. Поэтому когда линия связи передает трафик большого количества конечных узлов, в суммарном потоке пульсации сглаживаются и пропускная способность линии используется более рационально, без длительных простоев. Это эффект иллюстрируется рис. 3.7, на котором показаны неравномерные потоки пакетов, поступающие от конечных узлов 3, 4 и 10 в сети, изображенной на рис. 3.6. Предположим, что эти потоки передаются в направлении коммутатора 8, а следовательно,

накладываются друг на друга при прохождении линии связи между коммутаторами 5 и 8. Получающийся в результате суммарный поток является более равномерным, чем каждый из образующих его отдельных потоков.

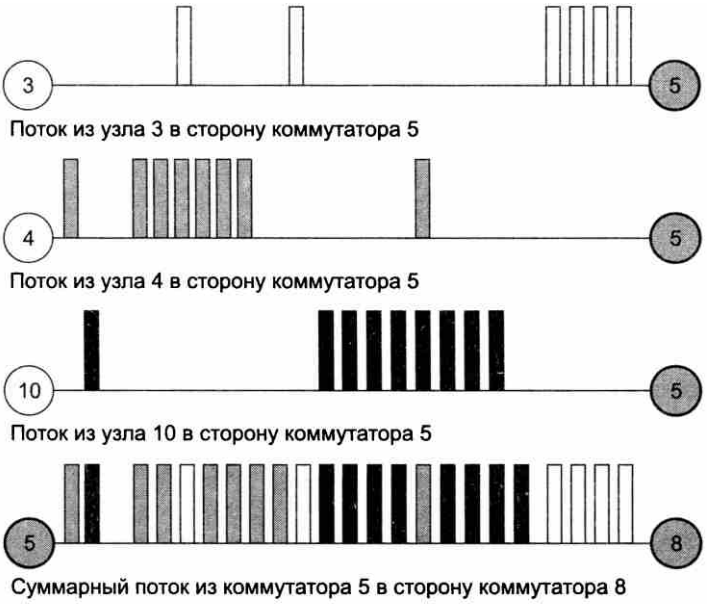


Рис. 3.7. Сглаживание трафика в сетях с коммутацией пакетов

Буферизация пакетов

Неопределенность и асинхронность перемещения данных в сетях с коммутацией пакетов предъявляет особые требования к работе коммутаторов в таких сетях.

Главное отличие пакетных коммутаторов[1] от коммутаторов в сетях с коммутацией каналов состоит в том, что они имеют внутреннюю **буферную память** для временного хранения пакетов.

Действительно, пакетный коммутатор не может принять решения о продвижении пакета, не имея в своей памяти всего пакета. Коммутатор проверяет контрольную сумму, и только если она говорит о том, что данные пакета не искажены, начинает обрабатывать пакет и по адресу назначения определяет следующий коммутатор. Поэтому *каждый* пакет последовательно, бит за битом, помещается во **входной буфер**. Имея в виду это свойство, говорят, что сети с коммутацией пакетов используют технику **сохранения с продвижением** (store-and-forward). Заметим, что для этой цели достаточно иметь буфер размером в один пакет.

[1] Для простоты будем далее называть коммутаторы сетей с коммутацией пакетов «пакетными коммутаторами», а сети с коммутацией пакетов — пакетными сетями.

Коммутатору нужны буферы *для согласования скоростей передачи данных в линиях связи*, подключенных к его интерфейсам. Действительно, если скорость поступления пакетов из одной линии связи в течение некоторого периода превышает пропускную способность той линии связи, в которую эти пакеты должны быть направлены, то во избежание потерь пакетов на целевом интерфейсе необходимо организовать выходную очередь (рис. 3.8).

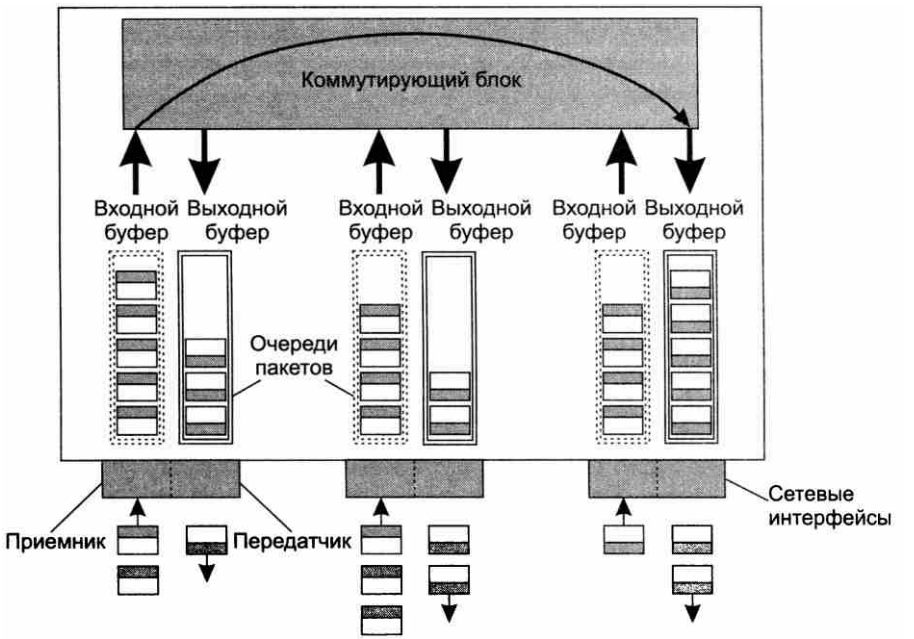


Рис. 3.8. Буферы и очереди пакетов в коммутаторе

Буферизация необходима пакетному коммутатору также *для согласования скорости поступления пакетов со скоростью их коммутации*. Если коммутирующий блок не успевает обрабатывать пакеты (анализировать заголовки и перебрасывать пакеты на нужный интерфейс), то на интерфейсах коммутатора возникают **входные очереди**. Очевидно, что для хранения входной очереди объем буфера должен превышать размер одного пакета. Существуют различные подходы к построению коммутирующего блока. Традиционный способ основан на одном центральном процессоре, который обслуживает все входные очереди коммутатора. Такой способ построения может приводить к большим очередям, так как производительность процессора разделяется между несколькими очередями. Современные способы построения коммутирующего блока основаны на многопроцессорном подходе, когда каждый интерфейс имеет свой встроенный процессор для обработки пакетов. Кроме того, существует центральный процессор, координирующий работу интерфейсных процессоров. Использование интерфейсных процессоров повышает производительность коммутатора и уменьшает очереди на входных интерфейсах. Однако такие очереди все равно могут возникать, так как центральный процессор по-прежнему остается «узким местом». Более подробно вопросы внутреннего устройства коммутаторов обсуждаются в главе 12.

Поскольку объем буферов в коммутаторах ограничен, иногда происходит потеря пакетов из-за переполнения буферов при временной перегрузке части сети, когда совпадают периоды пульсации нескольких информационных потоков. Для сетей с коммутацией пакетов потеря пакетов является обычным явлением, и для компенсации таких потерь в данной сетевой технологии предусмотрен ряд специальных механизмов, которые мы рассмотрим позже.

Пакетный коммутатор может работать на основании одного из трех методов продвижения пакетов:

- дейтаграммная передача;
- передача с установлением логического соединения;
- передача с установлением виртуального канала.

Дейтаграммная передача

Дейтаграммный способ передачи данных основан на том, что все передаваемые пакеты *продвигаются* (передаются от одного узла сети другому) *независимо* друг от друга на основании одних и тех же правил.

Процедура обработки пакета определяется только значениями параметров, которые он несет в себе, и текущим состоянием сети (например, в зависимости от ее нагрузки пакет может стоять в очереди на обслуживание большее или меньшее время). Однако никакая информация об *уже переданных пакетах* сетью не хранится и в ходе обработки очередного пакета во внимание не принимается. То есть каждый отдельный пакет рассматривается сетью как совершенно независимая единица передачи — **дейтаграмма**.

Решение о продвижении пакета принимается на основе таблицы коммутации[1], ставящей в соответствие адресам назначения пакетов информацию, однозначно определяющую следующий по маршруту транзитный (или конечный) узел. В качестве такой информации могут выступать идентификаторы интерфейсов данного коммутатора или адреса входных интерфейсов коммутаторов, следующих по маршруту.

На рис. 3.9 показана сеть, в которой шесть конечных узлов (*N*1–*N*6) связаны семью коммутаторами (*S*1–*S*7). Показаны также несколько перемещающихся по разным маршрутам пакетов с разными адресами назначения (*N*1–*N*6), на пути которых лежит коммутатор *S*1. При поступлении каждого из этих пакетов в коммутатор *S*1 выполняются просмотр соответствующей таблицы коммутации и выбор дальнейшего пути перемещения. Так, пакет с адресом *N*5 будет передан коммутатором *S*1 на интерфейс, ведущий к коммутатору *S*6, где в результате подобной процедуры этот пакет будет направлен конечному узлу-получателю *N*5.

---

[1] Напомним, что в разных технологиях для обозначения таблиц, имеющих упомянутое функциональное назначение, могут использоваться другие термины (таблица маршрутизации, таблица продвижения и др.).

Таблица коммутации коммутатора S1

Адрес назначения	Адрес следующего коммутатора
N1	Пакет не требуется передавать через сеть
N2	S2
N3	S3
N4	S3
N5	S6
N6	S6

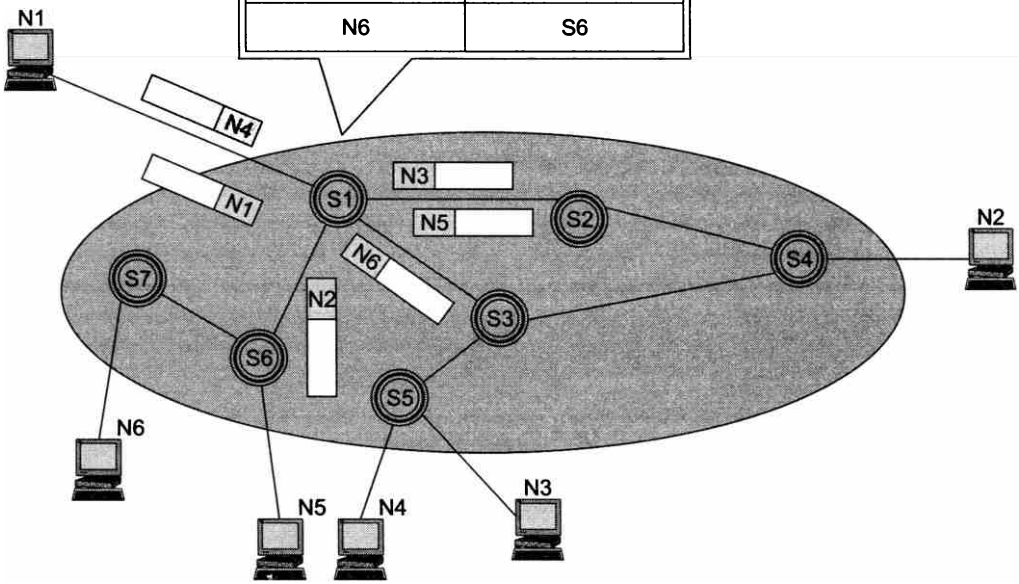


Рис. 3.9. Иллюстрация дейтаграммного принципа передачи пакетов

В таблице коммутации для одного и того же адреса назначения может содержаться несколько записей, указывающих соответственно на различные адреса следующего коммутатора. Такой подход называется **балансом нагрузки** и используется для повышения производительности и надежности сети. В примере, показанном на рис. 3.9, пакеты, поступающие в коммутатор *S*1 для узла назначения с адресом *N*2, в целях баланса нагрузки распределяются между двумя следующими коммутаторами — *S*2 и *S*3, что снижает нагрузку на каждый из них, а значит, сокращает очереди и ускоряет доставку. Некоторая «размытость» путей следования пакетов с одним и тем же адресом назначения через сеть является прямым следствием принципа независимой обработки каждого пакета, присущего дейтаграммному методу. Пакеты, следующие по одному и тому же адресу назначения, могут добираться до него разными путями также вследствие изменения состояния сети, например отказа промежуточных коммутаторов.

Дейтаграммный метод работает быстро, так как никаких предварительных действий перед отправкой данных проводить не требуется. Однако при таком методе трудно проверить факт доставки пакета узлу назначения. В этом методе доставка пакета не гарантируется, а выполняется по мере возможности — для описания такого свойства используется термин **доставка по возможности** (best effort).

Передача с установлением логического соединения

Следующий рассматриваемый нами способ продвижения пакетов основывается на знании устройствами сети «истории» обмена данными, например на запоминании узлом-отправителем числа отправленных, а узлом-получателем — числа полученных пакетов. Такого рода информация фиксируется в рамках логического соединения.

> Процедура согласования двумя конечными узлами сети некоторых параметров процесса обмена пакетами называется **установлением логического соединения**. Параметры, о которых договариваются два взаимодействующих узла, называются **параметрами логического соединения**.

Наличие логического соединения позволяет более рационально по сравнению с дейтаграммным способом обрабатывать пакеты. Например, при потере нескольких предыдущих пакетов может быть снижена скорость отправки последующих. Или благодаря нумерации пакетов и отслеживанию номеров отправленных и принятых пакетов можно повысить надежность путем отбрасывания дубликатов, упорядочивания поступивших и повторения передачи потерянных пакетов.

Параметры соединения могут быть: *постоянными*, то есть не изменяющимися в течение всего соединения (например, идентификатор соединения, способ шифрования пакета или максимальный размер поля данных пакета), или *переменными*, то есть динамически отражающими текущее состояние соединения (например, последовательные номера передаваемых пакетов).

Когда отправитель и получатель *фиксируют* начало нового соединения, они прежде всего «договариваются» о начальных значениях параметров процедуры обмена и только после этого начинают передачу собственно данных.

Передача с установлением соединения более надежна, но требует больше времени для передачи данных и вычислительных затрат от конечных узлов, что иллюстрирует рис. 3.10.

Процедура установления соединения состоит обычно из трех шагов.

1. Узел-инициатор соединения отправляет узлу-получателю служебный пакет с предложением установить соединение.
2. Если узел-получатель согласен с этим, то он посылает в ответ другой служебный пакет, подтверждающий установление соединения и предлагающий некоторые параметры, которые должны использоваться в рамках данного логического соединения. Это могут быть, например, идентификатор соединения, количество кадров, которые можно отправить без получения подтверждения, и т. п.

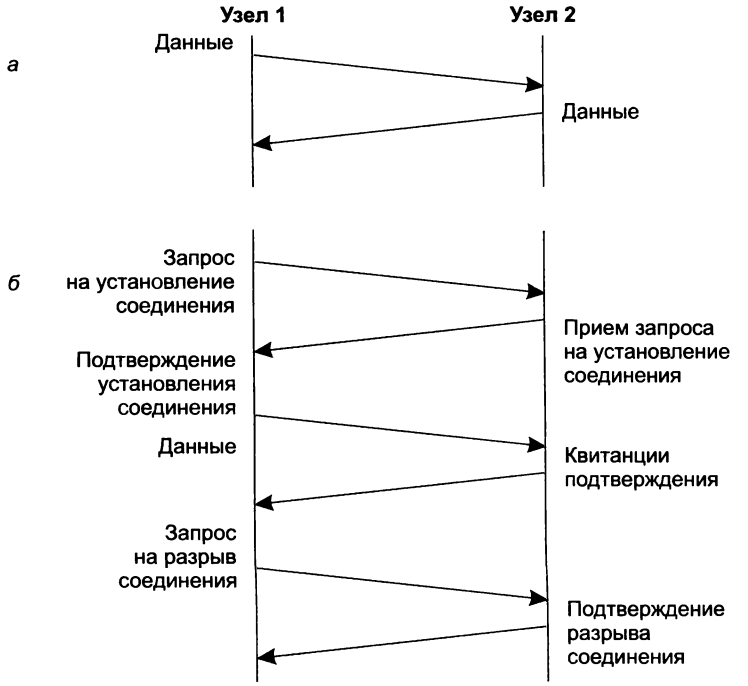


Рис. 3.10. Передача без установления соединения (*а*) и с установлением соединения (*б*)

3. Узел-инициатор соединения может закончить процесс установления соединения отправкой третьего служебного пакета, в котором сообщит, что предложенные параметры ему подходят.

Логическое соединение может быть рассчитано на передачу данных как в одном направлении — от инициатора соединения, так и в обоих направлениях. После передачи некоторого законченного набора данных, например определенного файла, узел-отправитель инициирует разрыв данного логического соединения, посылая соответствующий служебный кадр.

Заметим что, в отличие от передачи дейтаграммного типа, в которой поддерживается только один тип кадра — информационный, передача с установлением соединения должна поддерживать как минимум два типа кадров — информационные кадры переносят собственно пользовательские данные, а служебные предназначаются для установления (разрыва) соединения.

После того как соединение установлено и все параметры согласованы, конечные узлы начинают передачу собственно данных. Пакеты данных обрабатываются коммутаторами точно так же, как и при дейтаграммной передаче: из заголовков пакетов извлекаются адреса назначения и сравниваются с записями в таблицах коммутации, содержащих информацию о следующих шагах по маршруту. Так же как дейтаграммы, пакеты, относящиеся к одному логическому соединению, в некоторых случаях (например, при отказе линии связи) могут доставляться адресату по разным маршрутам.

Однако передача с установлением соединения имеет важное отличие от дейтаграммной передачи, поскольку в ней помимо обработки пакетов на коммутаторах имеет место *до-*

полнительная обработка пакетов на конечных узлах. Например, если при установлении соединения была оговорена передача данных в зашифрованном виде, то шифрование пакетов выполняется узлом-отправителем, а дешифрирование — узлом-получателем. Аналогично для обеспечения в рамках логического соединения надежности всю работу по нумерации пакетов, отслеживанию номеров доставленных и недоставленных пакетов, посылке копий и отбрасыванию дубликатов берут на себя конечные узлы.

ПРИМЕЧАНИЕ

Некоторые параметры логического соединения могут рассматриваться еще и как признаки информационного потока между узлами, установившими это логическое соединение.

Механизм установления логических соединений позволяет реализовывать дифференцированное обслуживание информационных потоков. Разное обслуживание могут получить даже потоки, относящиеся к одной и той же паре конечных узлов. Например, пара конечных узлов может установить два параллельно работающих логических соединения, в одном из которых передавать данные в зашифрованном виде, а в другом — открытым текстом.

Как видим, передача с установлением соединения предоставляет больше возможностей в плане надежности и безопасности обмена данными, чем дейтаграммная передача. Однако этот способ более медленный, так как он подразумевает дополнительные вычислительные затраты на установление и поддержание логического соединения.

Передача с установлением виртуального канала

Следующий способ продвижения данных основан на частном случае логического соединения, в число параметров которого входит жестко определенный для всех пакетов *маршрут*. То есть все пакеты, передаваемые в рамках данного соединения, должны проходить по одному и тому же закрепленному за этим соединением пути.

Единственный заранее проложенный фиксированный маршрут, соединяющий конечные узлы в сети с коммутацией пакетов, называют **виртуальным каналом** (virtual circuit, или virtual channel).

Виртуальные каналы прокладываются для *устойчивых* информационных потоков. С целью выделения потока данных из общего трафика каждый пакет этого потока помечается признаком особого вида — **меткой**.

Так же как в сетях с установлением логических соединений, прокладка виртуального канала начинается с отправки узлом-источником специального пакета — запроса на установление соединения. В запросе указываются адрес назначения и метка потока, для которого прокладывается этот виртуальный канал. Запрос, проходя по сети, формирует новую запись в каждом из коммутаторов, расположенных на пути от отправителя до получателя. Запись говорит о том, каким образом коммутатор должен обслуживать пакет, имеющий заданную метку. Образованный виртуальный канал идентифицируется той же меткой[1].

[1] Эта метка в различных технологиях называется по-разному: номером логического канала (Logical Channel Number, LCN) в технологии X.25, идентификатором соединения уровня канала данных (Data Link Connection Identifier, DLCI) в технологии Frame Relay, идентификатором виртуального канала (Virtual Channel Identifier, VCI) в технологии ATM.

После прокладки виртуального канала сеть может передавать по нему соответствующий поток данных. Во всех пакетах, которые переносят пользовательские данные, адрес назначения уже не указывается, его роль играет метка виртуального канала. При поступлении пакета на входной интерфейс коммутатор читает значение метки из заголовка пришедшего пакета и просматривает свою таблицу коммутации, по которой определяет, на какой выходной порт передать пришедший пакет.

На рис. 3.11 показана сеть, в которой проложено два виртуальных канала (Virtual Channel, VC), идентифицируемых метками VC1 и VC2. Первый проходит от конечного узла с адресом N1 до конечного узла с адресом N2 через промежуточные коммутаторы S1, S2 и S4. Второй виртуальный канал VC2 обеспечивает продвижение данных по пути *N*1-*S*1-*S*3-*S*5-*N*3. В общем случае между двумя конечными узлами может быть проложено несколько виртуальных каналов, например еще один виртуальный канал между узлами *N*1 и *N*2 мог бы проходить через промежуточный коммутатор *S*3. На рисунке показаны два пакета, несущие в своих заголовках метки потоков VC1 и VC2, которые играют роли адресов назначения.

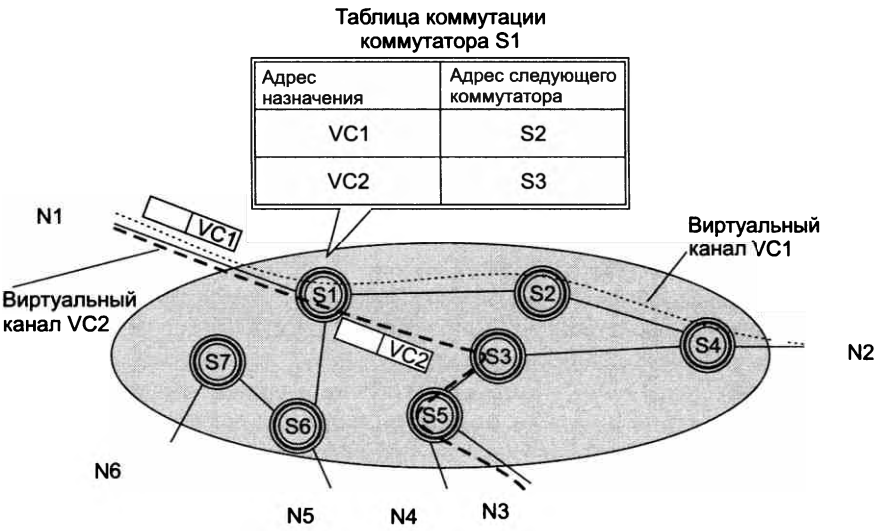


Рис. 3.11. Иллюстрация принципа работы виртуального канала

Таблица коммутации в сетях, использующих виртуальные каналы, отличается от таблицы коммутации в дейтаграммных сетях. Она содержит записи *только о проходящих через коммутатор виртуальных каналах*, а не обо всех возможных адресах назначения, как это имеет место в сетях с дейтаграммным алгоритмом продвижения. Обычно в крупной сети количество проложенных через узел виртуальных каналов существенно меньше общего количества узлов, поэтому и таблицы коммутации в этом случае намного короче, а следовательно, анализ такой таблицы занимает у коммутатора меньше времени. По той же причине метка короче адреса конечного узла и заголовок пакета в сетях с виртуальными каналами переносит по сети вместо длинного адреса компактный идентификатор потока.

ПРИМЕЧАНИЕ

Использование в сетях техники виртуальных каналов не делает их сетями с коммутацией каналов. Хотя в подобных сетях применяется процедура предварительного установления канала, этот канал является виртуальным, то есть по нему передаются отдельные пакеты, а не потоки информации с постоянной скоростью, как в сетях с коммутацией каналов.

В одной и той же сетевой технологии могут быть задействованы разные способы продвижения данных. Так, дейтаграммный протокол IP используется для передачи данных между отдельными сетями, составляющими Интернет. В то же время обеспечением надежной доставки данных между конечными узлами этой сети занимается протокол TCP, устанавливающий логические соединения без фиксации маршрута. И наконец, Интернет — это пример сети, применяющей технику виртуальных каналов, так как в состав Интернета входит немало сетей ATM и Frame Relay, поддерживающих виртуальные каналы.

Сравнение сетей с коммутацией пакетов и каналов

Прежде чем проводить техническое сравнение сетей с коммутацией пакетов и сетей с коммутацией каналов, проведем их неформальное сравнение на основе, как нам кажется, весьма продуктивной транспортной аналогии.

Транспортная аналогия для сетей с коммутацией пакетов и каналов

Для начала убедимся, что движение на дорогах имеет много общего с перемещением пакетов в сети *с коммутацией пакетов*.

Пусть автомобили в этой аналогии соответствуют пакетам, дороги — каналам связи, а перекрестки — коммутаторам. Подобно пакетам, автомобили перемещаются независимо друг от друга, разделяя пропускную способность дорог и создавая препятствия друг другу. Слишком интенсивный трафик, не соответствующий пропускной способности дороги, приводит к перегруженности дорог, в результате автомобили стоят в пробках, что соответствует очередям пакетов в коммутаторах.

На перекрестках происходит «коммутация» потоков автомобилей, каждый из автомобилей выбирает подходящее направление перекрестка, чтобы попасть в пункт назначения. Конечно, перекресток играет намного более пассивную роль по сравнению с коммутатором пакетов. Его активное участие в обработке трафика можно заметить только на регулируемых перекрестках, где светофор определяет очередность пересечения перекрестка потоками автомобилей. Еще активнее, естественно, поведение регулировщика трафика, который может выбрать для продвижения не только поток автомобилей в целом, но и отдельный автомобиль.

Как и в сетях с коммутацией пакетов, к образованию заторов на дорогах приводит неравномерность движения автомобилей. Так, даже кратковременное снижение скорости одного автомобиля на узкой дороге может создать большую пробку, которой бы не было, если бы все автомобили всегда двигались с одной и той же скоростью и равными интервалами.

А теперь попробуем найти общее у автомобильного движения и сетей с *коммутацией каналов*.

Иногда на дороге возникает ситуация, когда нужно обеспечить особые условия для движения колонны автомобилей. Например, представим, что очень длинная колонна автобусов перевозит детей из города в летний лагерь по многополосному шоссе. Для того чтобы колонна двигалась без препятствий, для ее движения заранее разрабатывается маршрут.

Затем на протяжении всего этого маршрута, который пересекает несколько перекрестков, для колонны выделяется отдельная полоса на всех отрезках шоссе. При этом полоса освобождается от другого трафика еще за некоторое время до начала движения колонны, и это резервирование отменяется только после того, как колонна достигает пункта назначения.

Во время движения все автомобили колонны едут с одинаковой скоростью и приблизительно равными интервалами между собой, не создавая препятствий друг другу. Очевидно, что для колонны автомобилей создаются наиболее благоприятные условия движения, но при этом автомобили теряют свою самостоятельность, превращаясь в поток, из которого нельзя «свернуть» в сторону. Дорога при такой организации движения используется нерационально, так как полоса простаивает значительную часть времени, как и полоса пропускания в сетях с коммутацией каналов.

Количественное сравнение задержек

Вернемся от автомобилей к сетевому трафику. Пусть пользователю сети необходимо передать достаточно неравномерный трафик, состоящий из периодов активности и пауз. Представим также, что он может выбрать, через какую сеть, с коммутацией каналов или пакетов, передавать свой трафик, причем в обеих сетях производительность каналов связи одинакова. Очевидно, что более эффективной с точки зрения временны́х затрат для нашего пользователя была бы работа в сети с коммутацией каналов, где ему в единоличное владение предоставляется зарезервированный канал связи. При этом способе все данные поступали бы адресату без задержки. Тот факт, что значительную часть времени зарезервированный канал будет простаивать (во время пауз), нашего пользователя не волнует — ему важно быстро решить собственную задачу.

Если бы пользователь обратился к услугам сети с коммутацией пакетов, то процесс передачи данных оказался бы более медленным, так как его пакеты, вероятно, не раз задерживались бы в очередях, ожидая освобождения необходимых сетевых ресурсов наравне с пакетами других абонентов.

Давайте рассмотрим более детально механизм возникновения задержек при передаче данных в сетях обоих типов. Пусть от конечного узла $N1$ отправляется сообщение к конечному узлу $N2$ (рис. 3.12). На пути передачи данных расположены два коммутатора.

В сети с коммутацией каналов данные после задержки, связанной с установлением канала, начинают передаваться на стандартной для канала скорости. Время доставки данных T адресату равно сумме *времени распространения сигнала в канале* t_{prg} и *времени передачи сообщения в канал* (называемом также *временем сериализации*) t_{trns}.

Наличие коммутаторов в сети с коммутацией каналов никак не влияет на суммарное время прохождения данных через сеть.

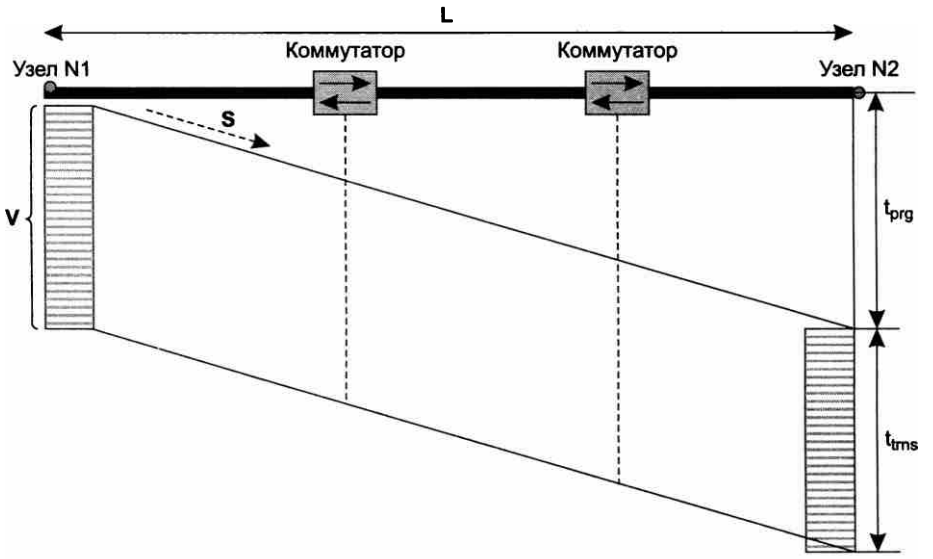


Рис. 3.12. Временна́я диаграмма передачи сообщения в сети с коммутацией каналов

ПРИМЕЧАНИЕ

Заметим, что время передачи сообщения в канал в точности совпадает со временем приема сообщения из канала в буфер узла назначения, то есть временем буферизации.

Время распространения сигнала зависит от расстояния между абонентами L и скорости S распространения электромагнитных волн в конкретной физической среде, которая колеблется от 0,6 до 0,9 скорости света в вакууме:

$$t_{\text{prg}} = L/S.$$

Время передачи сообщения в канал (а значит, и время буферизации в узле назначения) равно отношению объема сообщения V в битах к пропускной способности канала C в битах в секунду:

$$t_{\text{trna.}} = V/C.$$

В сети с коммутацией пакетов передача данных не требует обязательного установления соединения. Предположим, что в сеть, показанную на рис. 3.13, передается сообщение того же объема V, что и в предыдущем случае (см. рис. 3.12), однако оно разделено на пакеты, каждый из которых снабжен заголовком. Пакеты передаются от узла $N1$ узлу $N2$, между которыми расположены два коммутатора. На каждом коммутаторе каждый пакет изображен дважды: в момент прихода на входной интерфейс и в момент передачи в сеть с выходного интерфейса. Из рисунка видно, что коммутатор задерживает пакет на некоторое время. Здесь T_1 — время доставки адресату первого пакета сообщения, а T_{ps} — всего сообщения.

Сравнивая временны́е диаграммы передачи данных в сетях с коммутацией каналов и пакетов, отметим два факта:

- значения времени распространения сигнала (t_{prg}) в одинаковой физической среде на одно и то же расстояние одинаковы;
- учитывая, что значения пропускной способности каналов в обеих сетях одинаковы, значения времени передачи сообщения в канал (t_{trns}) будут *также равны*.

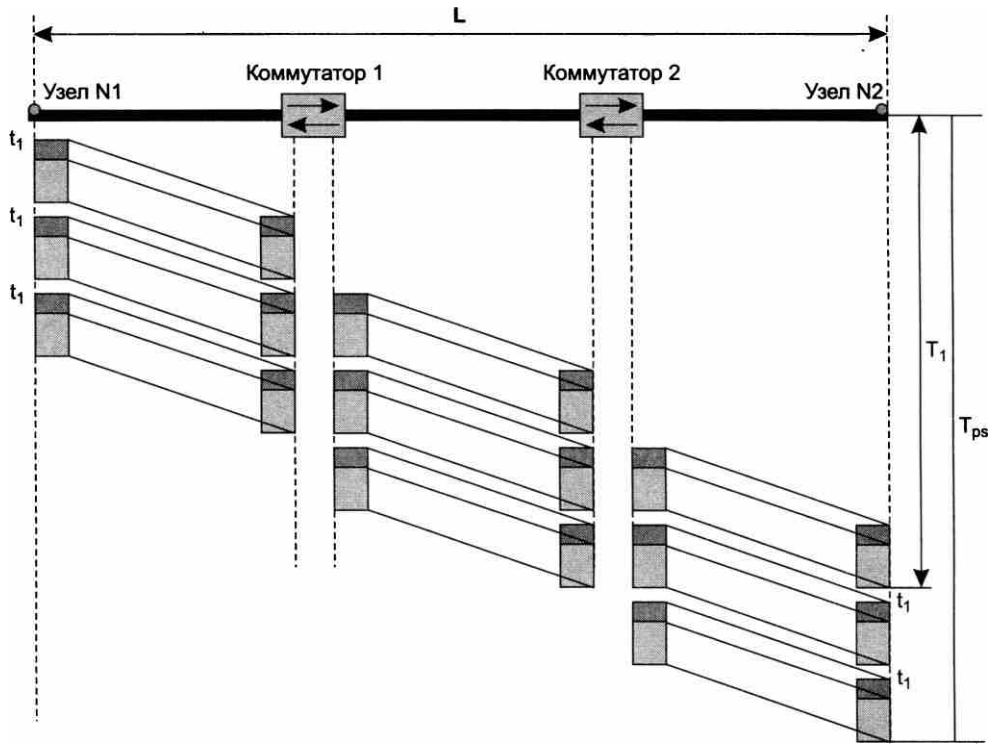


Рис. 3.13. Временна́я диаграмма передачи сообщения, разделенного на пакеты, в сети с коммутацией пакетов

Однако разбиение передаваемого сообщения на пакеты с последующей их передачей по сети с коммутацией пакетов приводит к дополнительным задержкам. Проследим путь первого пакета и отметим, из каких составляющих складывается время его передачи в узел назначения и какие из них специфичны для сети с коммутацией пакетов (рис. 3.14).

Время передачи одного пакета от узла *N*1 до коммутатора 1 можно представить в виде суммы нескольких слагаемых.

- Во-первых, время тратится в узле-отправителе *N*1:
  - t_1 — время формирования пакета, также называемое временем пакетизации (зависит от различных параметров работы программного и аппаратного обеспечения узла-отправителя и не зависит от параметров сети);
  - t_2 — время передачи в канал заголовка;
  - t_3 — время передачи в канал поля данных пакета.

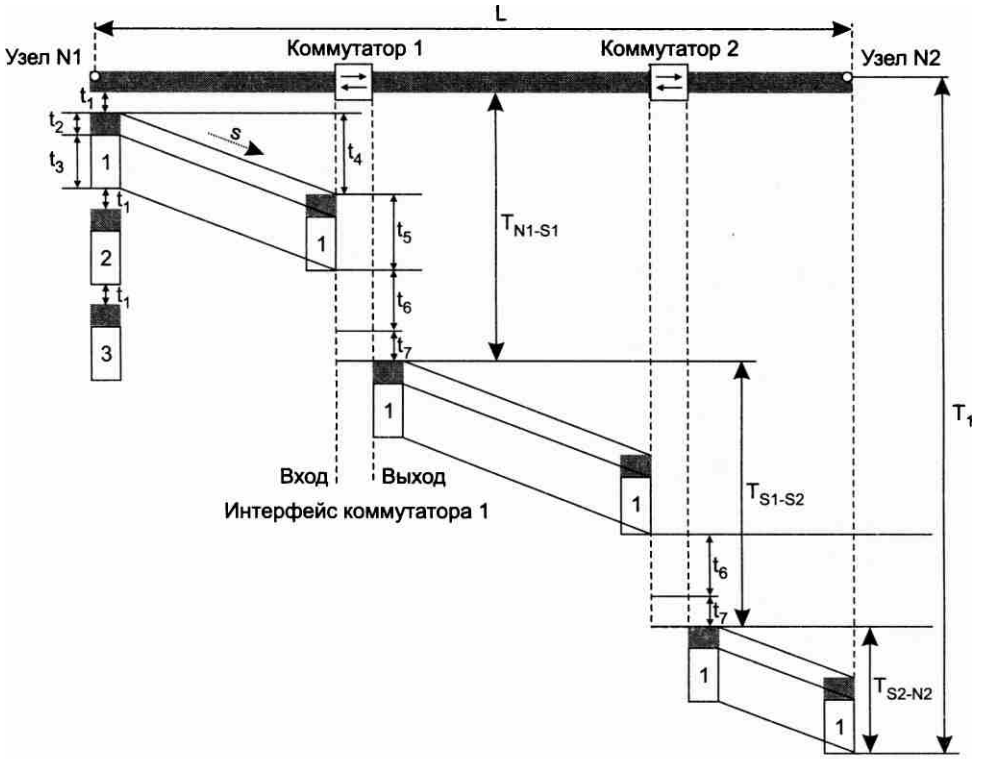


Рис. 3.14. Временна́я диаграмма передачи одного пакета в сети с коммутацией пакетов

- Во-вторых, дополнительное время тратится на распространение сигналов по каналам связи. Обозначим через t_4 время распространения сигнала, представляющего один бит информации, от узла $N1$ до коммутатора 1.
- В-третьих, дополнительное время тратится в промежуточном коммутаторе:
  - t_5 —время приема пакета с его заголовком из канала во входной буфер коммутатора; как уже было отмечено, это время равно (t_2 + t_3), то есть времени передачи пакета с заголовком в канал из узла источника;
  - t_6 — время ожидания пакета в очереди колеблется в очень широких пределах и заранее не известно, так как зависит от текущей загрузки сети;
  - t_7 — время коммутации пакета при его передаче в выходной порт фиксировано для конкретной модели и обычно невелико (от нескольких микросекунд до нескольких миллисекунд).

Обозначим через T_{N1-S1} время передачи пакета из узла $N1$ на выходной интерфейс коммутатора 1. Это время складывается из следующих составляющих:

$$T_{N1-S1} = t_1 + t_4 + t_5 + t_6 + t_7.$$

Обратите внимание, что среди слагаемых отсутствуют составляющие t_2 и t_3. Из рис. 3.14 видно, что передача битов из передатчика в канал совмещается по времени с передачей битов по каналу связи.

Время, затрачиваемое на оставшиеся два отрезка пути, обозначим соответственно T_{S1-S2} и T_{S2-N2}. Эти величины имеют такую же структуру, что и T_{N1-S1}, за исключением того, что в них не входит время пакетизации, и кроме того, T_{S2-N2} не включает время коммутации (так как отрезок заканчивается конечным узлом). Итак, полное *время передачи одного пакета* по сети составляет:

$$T1 = T_{N1-S1} + T_{S1-S2} + T_{S2-N2}.$$

А чему же будет равно время передачи сообщения, состоящего из нескольких пакетов? Сумме времен передачи каждого пакета? Конечно, нет! Ведь сеть с коммутацией пакетов работает как конвейер (см. рис. 3.13): пакет обрабатывается в несколько этапов, и все устройства сети выполняют эти этапы параллельно. Поэтому время передачи такого сообщения будет значительно меньше, чем сумма значений времени передачи каждого пакета сообщения. Точно рассчитать это время сложно из-за неопределенности состояния сети, и вследствие этого неопределенности значений времени ожидания пакетов в очередях коммутаторов. Однако если предположить, что пакеты стоят в очереди примерно одинаковое время, то общее *время передачи сообщения, состоящего из n пакетов*, можно оценить следующим образом:

$$T_{PS} = T_1 + (n - 1)(t_1 + t_5).$$

Пример

Используем для сравнения эффективности сетей с коммутацией каналов и пакетов пример (рис. 3.15). Два коммутатора объединены каналом связи с пропускной способностью 100 Мбит/с. Пользователи подключаются к сети с помощью каналов доступа (access link) с пропускной способностью 10 Мбит/с. Предположим, что все пользователи создают одинаковый пульсирующий трафик со средней скоростью 1 Мбит/с. При этом в течение непродолжительных периодов времени скорость данной предложенной нагрузки возрастает до максимальной скорости канала доступа, то есть до 10 Мбит/с. Такие периоды длятся не более 1 секунды. Предположим также, что все пользователи, подключенные к коммутатору *S*1, передают информацию только пользователям, подключенным к коммутатору *S*2.

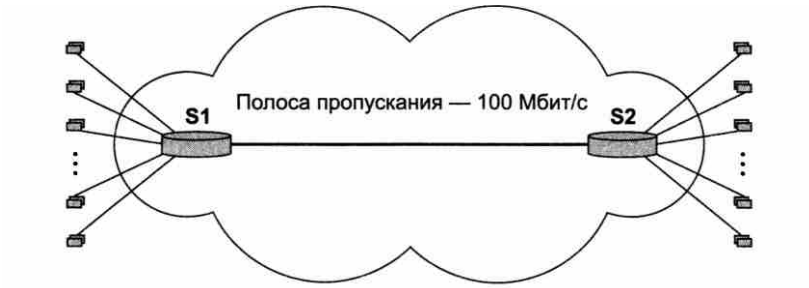


Рис. 3.15. Сравнение эффективности сетей с коммутацией пакетов и каналов

Пусть представленная на рисунке сеть является сетью с коммутацией каналов. Поскольку пики пользовательского трафика достигают 10 Мбит/с, каждому из пользователей необходимо установить соединение с пропускной способностью 10 Мбит/с. Таким образом, одновременно через сеть смогут передавать данные только 10 пользователей. Суммарная средняя скорость передачи данных через сеть будет равна только 10 Мбит/с (10 пользователей пере-

дают данные со средней скоростью 1 Мбит/). Следовательно, линия связи между коммутаторами хотя и имеет общую пропускную способность 100 Мбит/с, используется только на 10 %.

Теперь рассмотрим вариант, когда та же сеть работает на основе техники коммутации пакетов. При средней скорости пользовательских потоков 1 Мбит/с сеть может передавать одновременно до 100/1 = 100 (!) информационных потоков пользователей, полностью расходуя пропускную способность канала между коммутаторами. Однако это справедливо, только если емкости буферов коммутаторов достаточно для хранения пакетов на периодах перегрузки, когда суммарная скорость потока данных превышает 100 Мбит/с. Оценим необходимый объем буфера коммутатора *S*1. За период перегрузки в коммутатор *S*1 от каждого потока поступит 10 Мбит/с × 1 с = 10 Мбит, а от 100 потоков — 1000 Мбит. Из этих данных за 1 с коммутатор успеет передать в выходной канал только 100 Мбит. Значит, чтобы ни один пакет не был потерян во время перегрузки сети, общий объем входных буферов коммутатора должен быть не меньше 1000 – 100 = 900 Мбит, или более 100 Мбайт. Сегодняшние коммутаторы обычно имеют меньшие объемы буферов (1–10 Мбайт). Однако не нужно забывать, что вероятность совпадения периодов пиковой нагрузки у всех потоков, поступающих на входы коммутатора, очень мала. Так что даже если коммутатор имеет меньший объем буферной памяти, в подавляющем большинстве случаев он будет справляться с предложенной нагрузкой.

При сравнении сетей с коммутацией каналов и пакетов уместна аналогия с **мультипрограммными операционными системами**. Каждая отдельная программа в такой системе выполняется дольше, чем в однопрограммной системе, когда программе выделяется все процессорное время, пока она не завершит свое выполнение. Однако общее число программ, выполняемых в единицу времени, в мультипрограммной системе больше, чем в однопрограммной. Аналогично однопрограммной системе, в которой время от времени простаивают процессор или периферийные устройства, в сетях с коммутацией каналов при передаче пульсирующего трафика значительная часть зарезервированной пропускной способности каналов часто не используется.

Неопределенная пропускная способность сети с коммутацией пакетов — это плата за ее общую эффективность при некотором ущемлении интересов отдельных абонентов. Аналогично в мультипрограммной операционной системе время выполнения приложения предсказать заранее невозможно, так как оно зависит от количества других приложений, с которыми делит процессор данное приложение.

В заключение этого раздела приведем табл. 3.1, в которой сведены свойства обоих видов сетей. На основании этих данных можно аргументированно утверждать, в каких случаях рациональнее использовать сети с коммутацией каналов, а в каких — с коммутацией пакетов.

Таблица 3.1. Сравнение сетей с коммутацией каналов и пакетов

Коммутация каналов	Коммутация пакетов
Необходимо предварительно устанавливать соединение	Отсутствует этап установления соединения (дейтаграммный способ)
Адрес требуется только на этапе установления соединения	Адрес и другая служебная информация передаются с каждым пакетом
Сеть может отказать абоненту в установлении соединения	Сеть всегда готова принять данные от абонента
Гарантированная пропускная способность (полоса пропускания) для взаимодействующих абонентов	Пропускная способность сети для абонентов неизвестна, задержки передачи носят случайный характер

Коммутация каналов	Коммутация пакетов
Трафик реального времени передается без задержек	Ресурсы сети используются эффективно при передаче пульсирующего трафика
Высокая надежность передачи	Возможны потери данных из-за переполнения буферов
Нерациональное использование пропускной способности каналов, снижающее общую эффективность сети	Автоматическое динамическое распределение пропускной способности физического канала между абонентами

Ethernet — пример стандартной технологии с коммутацией пакетов

Рассмотрим, каким образом описанные ранее концепции воплощены в одной из первых стандартных сетевых технологий — технологии Ethernet, работающей с битовой скоростью 10 Мбит/с. В этом разделе мы коснемся только самых общих принципов функционирования Ethernet. Детальное описание технологии Ethernet вы найдете в части III.

- *Топология.* Существуют два варианта технологии Ethernet: Ethernet на разделяемой среде и коммутируемый вариант Ethernet. В первом случае все узлы сети разделяют общую среду передачи данных и сеть строится по топологии общей шины. На рис. 3.16 показан простейший вариант топологии — все компьютеры сети подключены к общей разделяемой среде, состоящей из одного сегмента коаксиального кабеля.

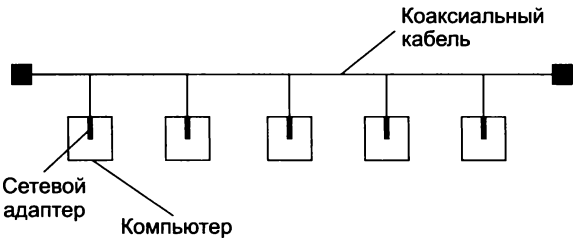


Рис. 3.16. Сеть Ethernet на разделяемой среде

В том случае, когда сеть Ethernet не использует разделяемую среду, а строится на коммутаторах, объединенных дуплексными каналами связи, говорят о коммутируемом варианте Ethernet. Топология в этом случае является топологией дерева, то есть такой, при которой между двумя любыми узлами сети существует ровно один путь. Пример топологии коммутируемой сети Ethernet показан на рис. 3.17.

Топологические ограничения (только древовидная структура связей коммутаторов) связаны со способом построения таблиц продвижения Ethernet-коммутаторами.

- *Способ коммутации.* В технологии Ethernet используется дейтаграммная коммутация пакетов. Единицы данных, которыми обмениваются компьютеры в сети Ethernet, называются кадрами. Кадр имеет фиксированный формат и наряду с полем данных содержит различную служебную информацию. В том случае, когда сеть Ethernet построена на коммутаторах, каждый коммутатор продвигает кадры в соответствии с теми

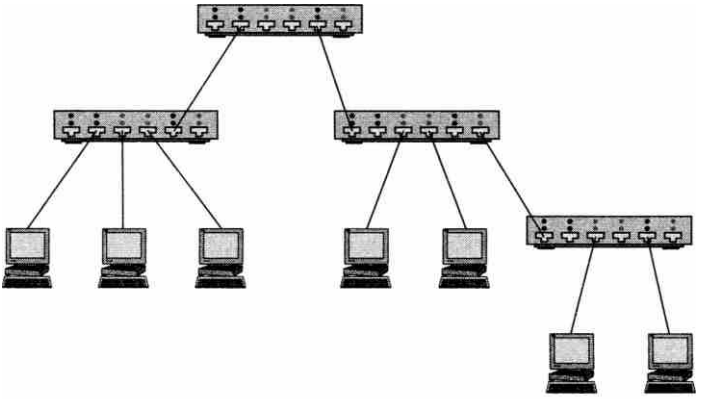


Рис. 3.17. Древовидная топология коммутируемой сети Ethernet

принципами коммутации пакетов, которые были описаны ранее. А вот в случае односегментной сети Ethernet возникает законный вопрос: где же выполняется коммутация? Где хотя бы один коммутатор, который, как мы сказали, является главным элементом любой сети с коммутацией пакетов? Или же Ethernet поддерживает особый вид коммутации? Оказывается, коммутатор в односегментной сети Ethernet существует, но его не так просто разглядеть, потому что его функции распределены по всей сети. В Ethernet такой «коммутатор» состоит из сетевых адаптеров и разделяемой среды. Сетевые адаптеры представляют собой интерфейсы этого виртуального коммутатора, а разделяемая среда — коммутационный блок, который передает кадры между интерфейсами. Часть функций коммутационного блока выполняют адаптеры, так как они решают, какой кадр адресован их компьютеру, а какой — нет.

- *Адресация*. Каждый компьютер, а точнее каждый сетевой адаптер, имеет уникальный аппаратный адрес (так называемый MAC-адрес, вы уже встречали этот акроним в главе 2). Ethernet-адрес является плоским числовым адресом, иерархия здесь не используется. Поддерживаются адреса для выборочной, широковещательной и групповой рассылки.
- *Разделение среды и мультиплексирование*. В сети Ethernet на коммутаторах каждый канал является дуплексным каналом связи, и проблемы его разделения между интерфейсами узлов не возникает. Передатчики Ethernet-коммутаторов используют дуплексные каналы связи для мультиплексирования потоков кадров от разных конечных узлов.

  В случае Ethernet на разделяемой среде конечные узлы применяют специальный метод доступа с целью синхронизации использования единственного полудуплексного канала связи, объединяющего все компьютеры сети. Единого арбитра в сети Ethernet на разделяемой среде нет, вместо этого все узлы прибегают к распределенному случайному методу доступа. Информационные потоки, поступающие от конечных узлов сети Ethernet, мультиплексируются в единственном передающем канале в режиме разделения времени. То есть кадрам разных потоков поочередно предоставляется канал. Чтобы подчеркнуть не всегда очевидную разницу между понятиями мультиплексирования и разделения среды, рассмотрим ситуацию, когда из всех компьютеров сети Ethernet только одному нужно передавать данные, причем данные нескольких приложений. В этом случае проблема разделения среды между сетевыми интерфейсами не возникает,

в то время как задача передачи нескольких информационных потоков по общей линии связи (то есть мультиплексирование) остается.

- *Кодирование*. Адаптеры в Ethernet работают с тактовой частотой 20 МГц, передавая в среду прямоугольные импульсы, соответствующие единицам и нулям данных компьютера. Когда начинается передача кадра, все его биты передаются в сеть с постоянной скоростью 10 Мбит/с (каждый бит передается за два такта). Эта скорость определяется пропускной способностью линии связи в сети Ethernet.
- *Надежность*. Для повышения надежности передачи данных в Ethernet используется стандартный прием — подсчет **контрольной суммы** и передача ее в концевике кадра. Если принимающий адаптер путем повторного подсчета контрольной суммы обнаруживает ошибку в данных кадра, то такой кадр отбрасывается. Повторная передача кадра протоколом Ethernet не выполняется, эта задача должна решаться другими технологиями, например протоколом TCP в сетях TCP/IP.
- *Очереди*. В коммутируемых сетях Ethernet очереди кадров, готовых к отправке, организуются обычным для сетей с коммутацией пакетов способом, то есть с помощью буферной памяти интерфейсов коммутатора.

  В сетях Ethernet на разделяемой среде коммутаторы отсутствуют. На первый взгляд может показаться, что в Ethernet на разделяемой среде нет очередей, свойственных сетям с коммутацией пакетов. Однако отсутствие коммутатора с буферной памятью в сети Ethernet не означает, что очередей в ней нет. Просто здесь очереди переместились в буферную память сетевого адаптера. В те периоды времени, когда среда занята передачей кадров других сетевых адаптеров, данные (предложенная нагрузка) по-прежнему поступают в сетевой адаптер. Так как они не могут быть переданы в это время в сеть, они начинают накапливаться во внутреннем буфере Ethernet-адаптера, образуя очередь. Поэтому в сети Ethernet существуют переменные задержки доставки кадров, как и во всех сетях с коммутацией пакетов.

Выводы

В сетях с коммутацией каналов по запросу пользователя создается непрерывный информационный канал, который образуется путем резервирования «цепочки» линий связи, соединяющих абонентов на время передачи данных. На всем своем протяжении канал передает данные с одной и той же скоростью. Это означает, что через сеть с коммутацией каналов можно качественно передавать данные, чувствительные к задержкам (голос, видео). Однако невозможность динамического перераспределения пропускной способности физического канала является принципиальным недостатком сети с коммутацией каналов, который делает ее неэффективной для передачи пульсирующего компьютерного трафика.

При коммутации пакетов передаваемые данные разбиваются в исходном узле на небольшие части — пакеты. Пакет снабжается заголовком, в котором указывается адрес назначения, поэтому он может быть обработан коммутатором независимо от остальных данных. Коммутация пакетов повышает производительность сети при передаче пульсирующего трафика, так как при обслуживании большого числа независимых потоков периоды их активности не всегда совпадают во времени. Пакеты поступают в сеть без предварительного резервирования ресурсов в том темпе, в котором их генерирует источник. Однако этот способ ком-

мутации имеет и отрицательные стороны: задержки передачи носят случайный характер, поэтому возникают проблемы при передаче трафика реального времени.

В сетях с коммутацией пакетов может использоваться один из трех алгоритмов продвижения пакетов: дейтаграммная передача, передача с установлением логического соединения и передача с установлением виртуального канала.

Контрольные вопросы

1. Трафик какого типа сеть с коммутацией каналов передает неэффективно?
2. Может ли сеть с коммутацией каналов работать без буферизации данных?
3. Из-за чего скорость передачи пользовательских данных в сетях с коммутацией пакетов всегда ниже пропускной способности каналов связи? Варианты ответов:
   а) из-за наличия заголовков у пакетов;
   б) из-за необходимости буферизовать пакеты перед обработкой;
   в) из-за низкого быстродействия маршрутизаторов.
4. Какие из сформулированных свойств составного канала всегда соответствуют действительности? Варианты ответов:
   а) данные, поступившие в составной канал, доставляются вызываемому абоненту без задержек;
   б) составной канал закрепляется за двумя абонентами на постоянной основе;
   в) количество элементарных каналов, входящих в составной канал между двумя абонентами, равно количеству промежуточных узлов плюс 1;
   г) составной канал имеет постоянную и фиксированную пропускную способность на всем своем протяжении.
5. Что является коммутатором в односегментой сети Ethernet на разделяемой среде? Варианты ответов:
   а) разделяемая среда;
   б) сетевые адаптеры;
   в) разделяемая среда и сетевые адаптеры.

ГЛАВА 4 Архитектура, стандартизация и классификация сетей

Декомпозиция задачи сетевого взаимодействия

Сетевая архитектура — это концептуальная схема функционирования компьютерной сети, определяющая принципы работы аппаратных и программных сетевых компонентов, организацию их связей, протоколы взаимодействия и способы физической передачи данных. Архитектура сети отражает декомпозицию общей задачи взаимодействия компьютеров на отдельные подзадачи, которые должны решаться отдельными компонентами сети — конечными узлами (компьютерами) и промежуточными узлами (коммутаторами и маршрутизаторами).

Многоуровневый подход

Для решения сложных задач, к которым относится и задача сетевого взаимодействия, используется известный универсальный прием — *декомпозиция*, то есть разбиение одной сложной задачи на несколько более простых задач-модулей. Декомпозиция состоит в четком определении функций каждого модуля, а также порядка их взаимодействия (то есть межмодульных интерфейсов). При таком подходе каждый модуль можно рассматривать как «черный ящик», абстрагируясь от его внутренних механизмов и концентрируя внимание на способе взаимодействия модулей. В результате такого логического упрощения задачи появляется возможность независимого тестирования, разработки и модификации модулей. Так, любой из показанных на рис. 4.1 модулей может быть переписан заново. Пусть, например, это будет модуль *A*, и если при этом разработчики сохранят без изменения межмодульные связи (в данном случае интерфейсы *A*-*B* и *A*-*C*), то это не потребует никаких изменений в остальных модулях.

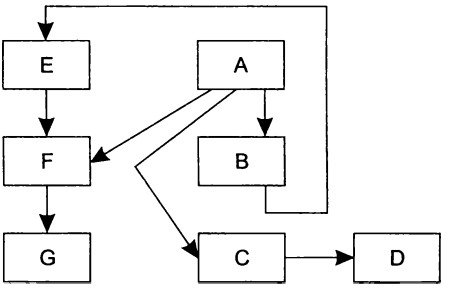


Рис. 4.1. Пример декомпозиции задачи

Еще более эффективной концепцией, развивающей идею декомпозиции, является *многоуровневый подход*. После представления исходной задачи в виде множества модулей эти модули группируют и упорядочивают по уровням, образующим иерархию. В соответствии с принципом иерархии для каждого промежуточного уровня можно указать непосредственно примыкающие к нему соседние вышележащий и нижележащий уровни (рис. 4.2).

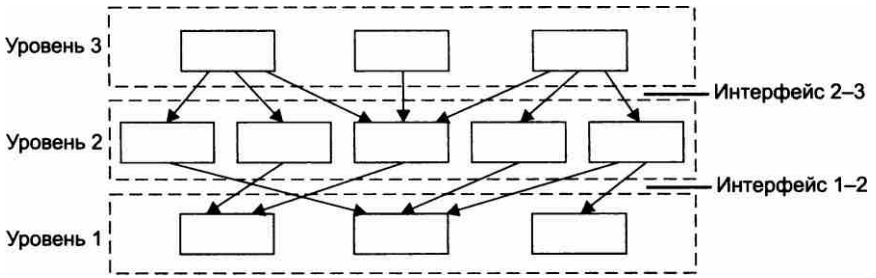


Рис. 4.2. Многоуровневый подход — создание иерархии задач

С одной стороны, группа модулей, составляющих каждый уровень, для решения своих задач должна обращаться с запросами только к модулям соседнего нижележащего уровня. С другой стороны, результаты работы каждого из модулей, отнесенных к некоторому уровню, могут быть переданы только модулям соседнего вышележащего уровня. Такая иерархическая декомпозиция задачи предполагает четкое определение функций и интерфейсов не только отдельных модулей, но и каждого уровня.

Межуровневый интерфейс, называемый также **интерфейсом услуг**, определяет набор функций, которые нижележащий уровень предоставляет вышележащему (рис. 4.3).

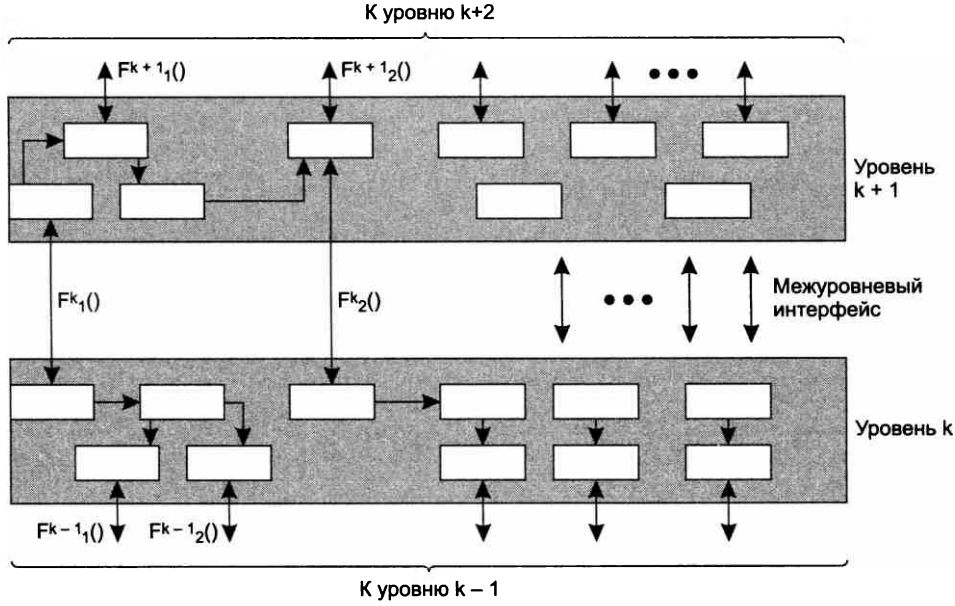


Рис. 4.3. Концепция многоуровневого взаимодействия

Такой подход дает возможность проводить разработку, тестирование и модификацию отдельного уровня независимо от других уровней. Иерархическая декомпозиция позволяет, двигаясь от более низкого уровня к более высокому, переходить ко все более и более абстрактному, а значит, более простому представлению исходной задачи.

Пример

Рассмотрим задачу считывания логической записи из файла, расположенного на локальном диске. Ее (очень упрощенно) можно представить в виде следующей иерархии частных задач.

1. Поиск по символьному имени файла его характеристик, необходимых для доступа к данным: информации о физическом расположении файла на диске, размер и др.

Поскольку функции этого уровня связаны только с просмотром каталогов, представления о файловой системе на этом уровне чрезвычайно абстрактны: файловая система имеет вид графа, в узлах которого находятся каталоги, а листьями являются файлы. Никакие детали физической и логической организации данных на диске данный уровень не интересуют.

2. Определение считываемой части файла.

Для решения этой задачи необходимо снизить степень абстракции файловой системы. Функции данного уровня оперируют файлом как совокупностью определенным образом связанных физических блоков диска.

3. Считывание данных с диска.

После определения номера физического блока файловая система обращается к системе ввода-вывода для выполнения операции чтения. На этом уровне уже фигурируют такие детали устройства файловой системы, как номера цилиндров, дорожек, секторов.

Среди функций, которые может запросить прикладная программа, обращаясь к верхнему уровню файловой системы, может быть, например, такая:

```
ПРОЧИТАТЬ 22 ЛОГИЧЕСКУЮ ЗАПИСЬ ФАЙЛА DIR1/MY/FILE.TXT
```

Верхний уровень не может выполнить этот запрос «только своими силами», определив по символьному имени `DIR1/MY/FILE.TXT` физический адрес файла, он обращается с запросом к нижележащему уровню:

```
ПРОЧИТАТЬ 22 ЛОГИЧЕСКУЮ ЗАПИСЬ ИЗ ФАЙЛА,
ИМЕЮЩЕГО ФИЗИЧЕСКИЙ АДРЕС 174 И РАЗМЕР 235
```

В ответ на запрос второй уровень определяет, что файл с адресом 174 занимает на диске пять несмежных областей, а искомая запись находится в четвертой области в физическом блоке 345. После этого он обращается к драйверу с запросом о чтении требуемой логической записи.

В соответствии с этой упрощенной схемой взаимодействие уровней файловой системы является однонаправленным — сверху вниз. Однако реальная картина существенно сложнее. Действительно, чтобы определить характеристики файла, верхний уровень должен «раскрутить» его символьное имя, то есть последовательно прочитать всю цепочку каталогов, указанную в имени файла. А это значит, что для решения своей задачи он несколько раз обратится к нижележащему уровню, который, в свою очередь, несколько раз «попросит» драйвер считать данные каталогов с диска. И каждый раз результаты будут передаваться снизу вверх.

Задача организации взаимодействия компьютеров в сети тоже может быть представлена в виде иерархически организованного множества модулей. Например, модулям нижнего уровня можно поручить вопросы, связанные с надежной передачей информации между двумя соседними узлами, а модулям следующего, более высокого уровня — транспорти-

ровку сообщений в пределах всей сети. Очевидно, что последняя задача — организация связи двух любых, не обязательно соседних, узлов — является более общей, и поэтому ее решение может быть получено путем многократных обращений к нижележащему уровню. Так, организация взаимодействия узлов *A* и *B* может быть сведена к поочередному взаимодействию пар промежуточных смежных узлов (рис. 4.4).

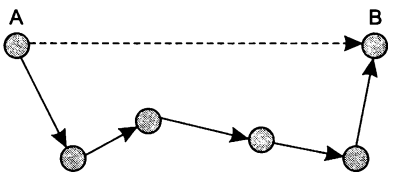


Рис. 4.4. Взаимодействие произвольной пары узлов

Протокол и стек протоколов

Многоуровневое представление средств сетевого взаимодействия имеет свою специфику, связанную с тем, что в процессе обмена сообщениями участвуют по меньшей мере *две стороны*, то есть в данном случае необходимо организовать согласованную работу двух иерархий аппаратных и программных средств на разных компьютерах. Оба участника сетевого обмена должны принять множество соглашений. Например, они должны согласовать уровни и форму электрических сигналов, способ определения размера сообщений, договориться о методах контроля достоверности и т. п. Другими словами, соглашения должны быть приняты на всех уровнях, начиная от самого низкого — уровня передачи битов, и заканчивая самым высоким, реализующим обслуживание пользователей сети.

На рис. 4.5 показана модель взаимодействия двух узлов. С каждой стороны средства взаимодействия представлены четырьмя уровнями. Каждый уровень поддерживает интерфейсы двух типов. Во-первых, это интерфейсы услуг с выше- и нижележащим уровнями «своей» иерархии средств. Во-вторых, это интерфейс со средствами взаимодействия другой стороны, расположенными на том же уровне иерархии. Этот тип интерфейса называют **протоколом**. Таким образом, протокол всегда является одноранговым интерфейсом.

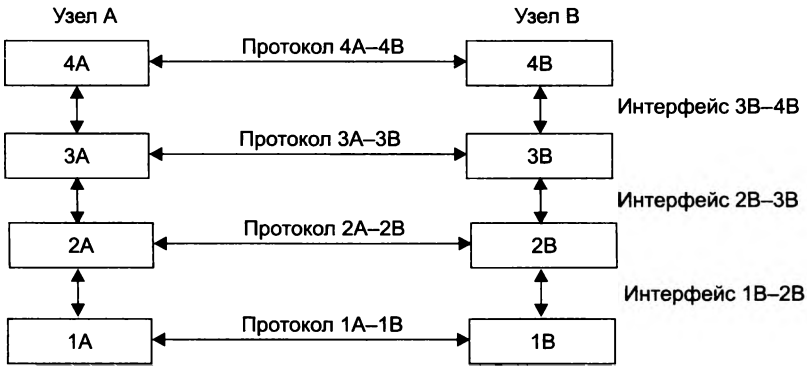


Рис. 4.5. Взаимодействие двух узлов

В сущности, термины «протокол» и «интерфейс» выражают одно и то же понятие — формализованное описание процедуры взаимодействия двух объектов, но традиционно в сетях за ними закрепили разные области действия: протоколы определяют правила взаимодействия модулей одного уровня в разных узлах, а интерфейсы — правила взаимодействия модулей соседних уровней в одном узле.

Иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети, называется **стеком протоколов**.

Протоколы нижних уровней часто реализуются комбинацией программных и аппаратных средств, а протоколы верхних уровней, как правило, программными средствами.

Программный модуль, реализующий некоторый протокол, называют протокольной сущностью, или, для краткости, тоже протоколом. Понятно, что один и тот же протокол может быть реализован с разной степенью эффективности. Именно поэтому при сравнении протоколов следует учитывать не только логику их работы, но и качество программной реализации. Более того, на эффективность взаимодействия устройств в сети влияет качество всей совокупности протоколов, составляющих стек, в частности то, *насколько рационально распределены функции между протоколами* разных уровней и насколько хорошо определены интерфейсы между ними.

Протокольные сущности одного уровня двух взаимодействующих сторон обмениваются сообщениями в соответствии с определенным для них протоколом. Сообщения состоят из заголовка и поля данных (иногда оно может отсутствовать). Обмен сообщениями является своеобразным языком общения, с помощью которого каждая из сторон «объясняет» другой стороне, что необходимо сделать на каждом этапе взаимодействия. Работа каждого протокольного модуля состоит в интерпретации заголовков поступающих к нему сообщений и выполнении связанных с этим действий. Заголовки сообщений разных протоколов имеют разную структуру, что соответствует различиям в их функциональности. Понятно, что чем сложнее структура заголовка сообщения, тем более сложные функции возложены на соответствующий протокол.

Модель OSI

Из того что протокол является соглашением, принятым двумя взаимодействующими узлами сети, совсем не следует, что он обязательно является стандартным. Но на практике при реализации сетей стремятся использовать стандартные протоколы. Это могут быть фирменные, национальные или международные стандарты.

В начале 80-х годов ряд международных организаций по стандартизации, в частности International Organization for Standardization (ISO), часто называемая International Standards Organization, а также International Telecommunications Union (ITU) и некоторые другие, разработали стандартную модель **взаимодействия открытых систем** (Open System Interconnection, OSI). Эта модель сыграла значительную роль в развитии компьютерных сетей.

Общая характеристика модели OSI

К концу 70-х годов в мире уже существовало большое количество фирменных стеков коммуникационных протоколов, среди которых можно назвать, например, такие популярные

стеки, как DECnet, TCP/IP и IBM SNA. Подобное разнообразие средств межсетевого взаимодействия вывело на первый план проблему несовместимости устройств, использующих разные протоколы. Одним из путей разрешения этой проблемы в то время виделся всеобщий переход на единый, общий для всех систем стек протоколов, созданный с учетом недостатков уже существующих стеков. Такой академический подход к созданию нового стека начался с разработки модели OSI и занял семь лет (с 1977 по 1984 год). Назначение модели OSI состоит в обобщенном представлении средств сетевого взаимодействия. Она разрабатывалась в качестве своего рода универсального языка сетевых специалистов, именно поэтому ее называют справочной моделью.

ВНИМАНИЕ

Модель OSI определяет, во-первых, уровни взаимодействия систем в сетях с коммутацией пакетов, во-вторых, стандартные названия уровней, в-третьих, функции, которые должен выполнять каждый уровень. Модель OSI не содержит описаний реализаций конкретного набора протоколов.

В модели OSI средства взаимодействия делятся на *семь* уровней: прикладной, представления, сеансовый, транспортный, сетевой, канальный и физический (рис. 4.6). Каждый уровень связан с совершенно определенным аспектом взаимодействия сетевых устройств.

ВНИМАНИЕ

Модель OSI описывает только системные средства взаимодействия, реализуемые операционной системой, системными утилитами, системными аппаратными средствами. Модель не включает средства взаимодействия приложений конечных пользователей. Важно различать уровень взаимодействия приложений и прикладной уровень семиуровневой модели.

Приложения могут реализовывать собственные протоколы взаимодействия, используя для этих целей многоуровневую совокупность системных средств. Именно для этого в распоряжение программистов предоставляется **прикладной программный интерфейс** (Application Program Interface, API). В соответствии с идеальной схемой модели OSI приложение может обращаться с запросами только к самому верхнему уровню — прикладному, однако на практике многие стеки коммуникационных протоколов предоставляют возможность программистам напрямую обращаться к сервисам, или службам, нижележащих уровней.

Например, некоторые СУБД имеют встроенные средства удаленного доступа к файлам. При наличии этих средств приложение, выполняя доступ к удаленным ресурсам, не использует системную файловую службу; оно обходит верхние уровни модели OSI и обращается непосредственно к ответственным за транспортировку сообщений по сети системным средствам, которые располагаются на нижних уровнях модели OSI.

Итак, пусть приложение узла *A* хочет взаимодействовать с приложением узла *B*. Для этого приложение *A* обращается с запросом к прикладному уровню, например к файловой службе. На основании этого запроса программное обеспечение прикладного уровня формирует сообщение стандартного формата. Но для того чтобы доставить эту информацию по назначению, предстоит решить еще много задач, ответственность за которые несут нижележащие уровни.

После формирования сообщения прикладной уровень направляет его вниз по стеку уровню представления. Протокол уровня представления на основании информации, полученной из заголовка сообщения прикладного уровня, выполняет требуемые действия и добавляет

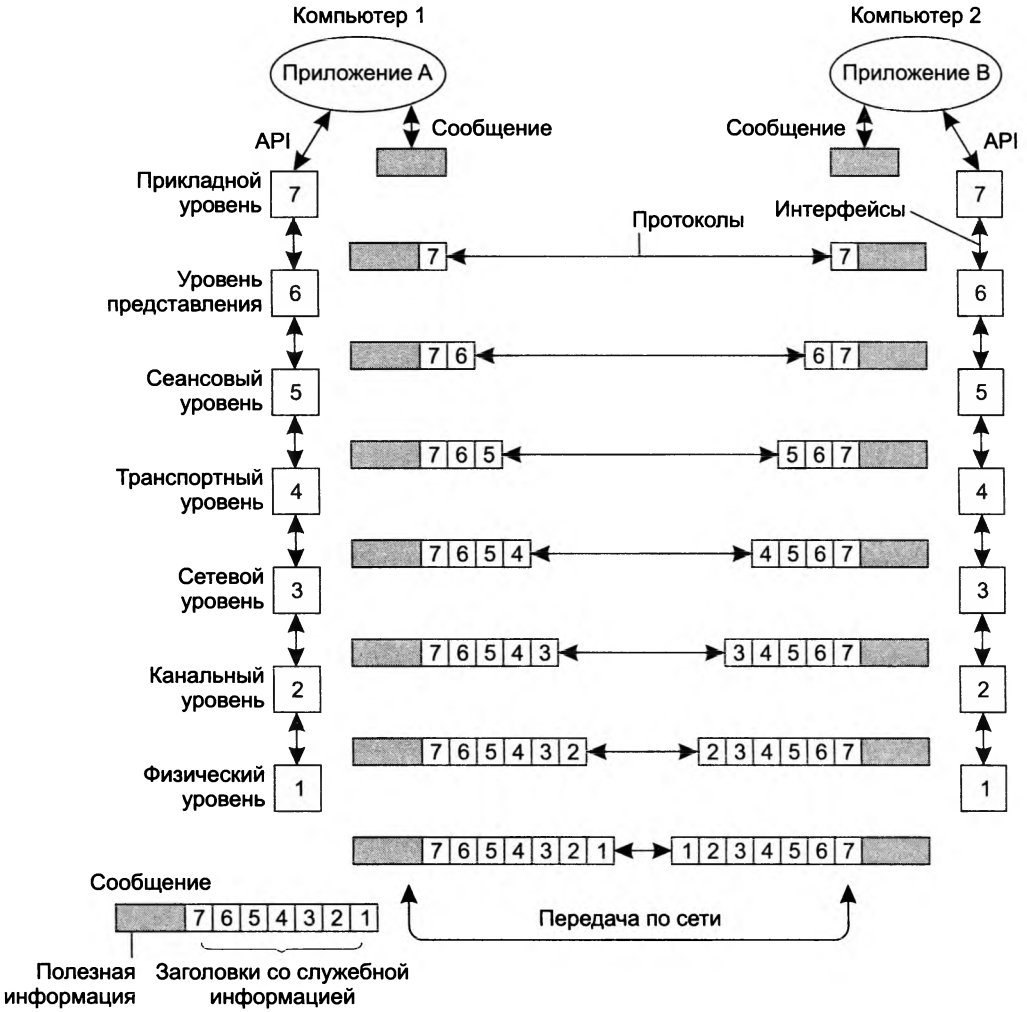


Рис. 4.6. Модель взаимодействия открытых систем ISO/OSI

к сообщению собственную служебную информацию — заголовок уровня представления, в котором содержатся указания для протокола уровня представления машины-адресата. Полученное в результате сообщение передается вниз сеансовому уровню, который, в свою очередь, добавляет свой заголовок, и т. д. (Некоторые реализации протоколов помещают служебную информацию не только в начале сообщения в виде заголовка, но и в конце в виде так называемого концевика.) Наконец, сообщение достигает нижнего, физического, уровня, который, собственно, и передает его по линиям связи машине-адресату. К этому моменту сообщение «обрастает» заголовками всех уровней (рис. 4.7).

Физический уровень помещает сообщение на физический выходной интерфейс компьютера 1, и оно начинает свое «путешествие» по сети (до этого момента сообщение передавалось от одного уровня другому в пределах компьютера 1).

Рис. 4.7. Вложенность сообщений различных уровней

Когда сообщение по сети поступает на входной интерфейс компьютера 2, оно принимается его физическим уровнем и последовательно перемещается вверх с уровня на уровень. Каждый уровень анализирует и обрабатывает заголовок своего уровня, выполняя соответствующие функции, а затем удаляет этот заголовок и передает сообщение вышележащему уровню.

Как видно из описания, протокольные сущности одного уровня не общаются между собой непосредственно, в этом общении всегда участвуют посредники — средства протоколов нижележащих уровней. И только физические уровни различных узлов взаимодействуют непосредственно.

В стандартах ISO для обозначения единиц обмена данными, с которыми имеют дело протоколы разных уровней, используется общее название **протокольная единица данных** (Protocol Data Unit, PDU). Для обозначения единиц обмена данными конкретных уровней часто используются *специальные* названия, в частности: **сообщение**, **кадр**, **пакет**, **дейтаграмма**, **сегмент**.

Физический уровень

Физический уровень (physical layer) имеет дело с передачей потока битов по физическим каналам связи, таким как коаксиальный кабель, витая пара, оптоволоконный кабель или цифровой территориальный канал. Функции физического уровня реализуются на всех устройствах, подключенных к сети. Со стороны компьютера функции физического уровня выполняются сетевым адаптером или последовательным портом.

Примером протокола физического уровня может служить спецификация 1000Base-T технологии Ethernet, которая определяет в качестве используемого кабеля неэкранированную витую пару категории 5 с волновым сопротивлением 100 Ом, разъем RJ-45, максимальную длину физического сегмента 100 метров, манчестерский код для представления данных в кабеле, а также некоторые другие характеристики среды и электрических сигналов.

Физический уровень не вникает в смысл информации, которую он передает. Для него эта информация представляет собой однородный поток битов, которые нужно доставить без искажений и в соответствии с заданной тактовой частотой (интервалом между соседними битами).

Канальный уровень

Канальный уровень (data link layer) обеспечивает прозрачность соединения для сетевого уровня. Для этого он предлагает ему следующие услуги:

- установление логического соединения между взаимодействующими узлами;
- согласование в рамках соединения скоростей передатчика и приемника информации;
- обеспечение надежной передачи, обнаружение и коррекция ошибок.

Для решения этих задач канальный уровень формирует из пакетов собственные протокольные единицы данных — **кадры**, состоящие из поля данных и заголовка. Канальный уровень помещает пакет в поле данных одного или нескольких кадров и заполняет собственной служебной информацией заголовок кадра.

В сетях, построенных на основе разделяемой среды, канальный уровень выполняет еще одну функцию — проверяет доступность разделяемой среды. Эту функцию иногда выделяют в отдельный подуровень **управления доступом к среде** (Medium Access Control, MAC).

Протокол канального уровня обычно работает в пределах сети, являющейся одной из частей более крупной составной сети, объединенной протоколами сетевого уровня. Адреса, с которыми работает протокол канального уровня, используются для доставки кадров только в пределах этой сети, а для перемещения пакетов между сетями применяются адреса уже следующего, сетевого уровня.

> Протоколы канального уровня реализуются как на конечных узлах (средствами сетевых адаптеров и их драйверов), так и на всех промежуточных сетевых устройствах.

Рассмотрим более подробно работу канального уровня начиная с момента, когда сетевой уровень отправителя передает канальному уровню пакет, а также указание, какому узлу его передать. Для решения этой задачи канальный уровень создает кадр, который имеет поле данных и заголовок. Канальный уровень помещает (*инкапсулирует*) пакет в поле данных кадра и заполняет соответствующей служебной информацией заголовок кадра. Важнейшей информацией заголовка кадра является адрес назначения, на основании которого коммутаторы сети будут продвигать пакет.

Одной из задач канального уровня является *обнаружение и коррекция ошибок*. Канальный уровень может обеспечить надежность передачи, например путем фиксирования границ кадра, помещая специальную последовательность битов в его начало и конец, а затем добавляя к кадру контрольную сумму. Контрольная сумма вычисляется по некоторому алгоритму как функция от всех байтов кадра. На стороне получателя канальный уровень группирует биты, поступающие с физического уровня, в кадры, снова вычисляет контрольную сумму полученных данных и сравнивает результат с контрольной суммой, переданной в кадре. Если они совпадают, кадр считается правильным. Если же контрольные суммы не совпадают, фиксируется ошибка.

В функции канального уровня входит не только обнаружение ошибок, но и их исправление за счет повторной передачи поврежденных кадров. Однако эта функция не является обязательной, и в некоторых реализациях канального уровня она отсутствует, например в Ethernet.

Прежде чем переправить кадр физическому уровню для непосредственной передачи данных в сеть, канальному уровню может потребоваться решить еще одну важную задачу.

Если в сети используется разделяемая среда, то прежде чем физический уровень начнет передавать данные, канальный уровень должен *проверить доступность среды*. Как уже отмечалось, функции проверки доступности разделяемой среды иногда выделяют в отдельный подуровень управления доступом к среде (подуровень MAC).

Если разделяемая среда освободилась (когда она не используется, то такая проверка, конечно, пропускается), кадр передается средствами физического уровня в сеть, проходит по каналу связи и поступает в виде последовательности битов в распоряжение физического уровня узла назначения. Этот уровень, в свою очередь, передает полученные биты «наверх» канальному уровню своего узла.

Протокол канального уровня обычно работает в пределах сети, входящей в виде одной из составляющих в более крупную составную сеть, объединенную протоколами сетевого уровня. Адреса, с которыми работает протокол канального уровня, используются для доставки кадров только в пределах этой сети, а для перемещения пакетов между сетями применяются уже адреса следующего, сетевого, уровня.

В локальных сетях канальный уровень поддерживает весьма мощный и законченный набор функций по пересылке сообщений между узлами сети. В некоторых случаях протоколы канального уровня локальных сетей оказываются самодостаточными транспортными средствами и могут допускать работу непосредственно поверх себя протоколов прикладного уровня или приложений без привлечения средств сетевого и транспортного уровней. Тем не менее для качественной передачи сообщений в сетях с произвольной топологией функций канального уровня оказывается недостаточно.

Сетевой уровень

Сетевой уровень (network layer) служит для образования единой транспортной системы, объединяющей несколько сетей и называемой **составной сетью**, или **интернетом**[1].

> Технология, позволяющая соединять в единую сеть множество сетей, в общем случае построенных на основе разных технологий, называется технологией **межсетевого взаимодействия** (internetworking).

На рис. 4.8 показано несколько сетей, каждая из которых использует собственную технологию канального уровня: Ethernet, FDDI, Token Ring, ATM, Frame Relay. На базе этих технологий любая из указанных сетей может связывать между собой любых пользователей, но только *своей* сети, и не способна обеспечить передачу данных в другую сеть. Причина такого положения вещей очевидна и кроется в существенных отличиях одной технологии от другой. Даже наиболее близкие технологии LAN — Ethernet, FDDI, Token Ring, имеющие одну и ту же систему адресации (адреса подуровня MAC, называемые MAC-адресами), отличаются друг от друга форматом используемых кадров и логикой работы протоколов. Еще больше отличий между технологиями LAN и WAN. Во многих технологиях WAN задействована техника предварительно устанавливаемых виртуальных каналов, иденти-

---

[1] Не следует путать интернет (со строчной буквы) с Интернетом (с прописной буквы). Интернет — это самая известная и охватывающая весь мир реализация составной сети, построенная на основе технологии TCP/IP.

фикаторы которых применяются в качестве адресов. Все технологии имеют собственные форматы кадров (в технологии ATM кадр даже называется иначе — ячейкой) и, конечно, собственные стеки протоколов.

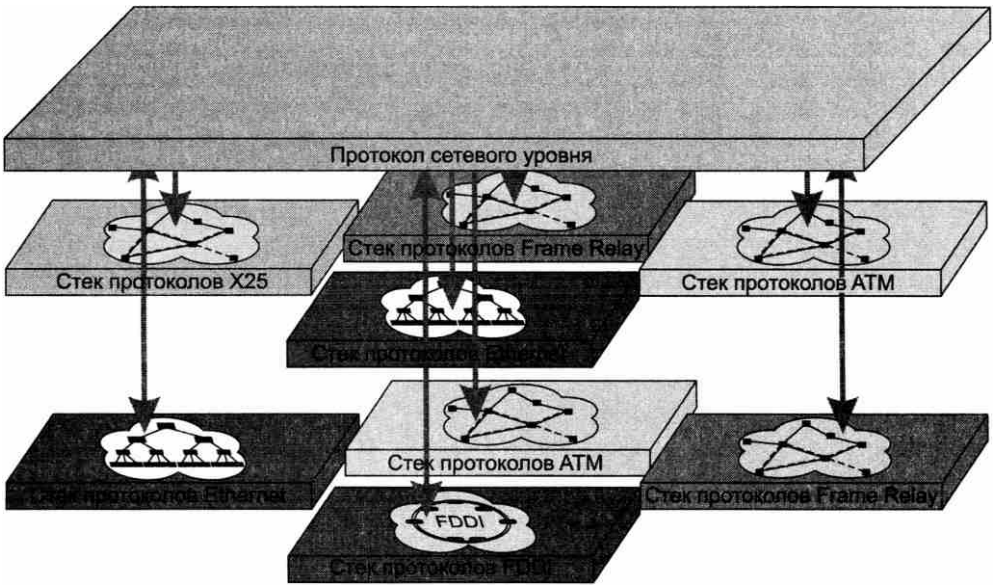


Рис. 4.8. Иллюстрация необходимости сетевого уровня

Чтобы связать между собой сети, построенные на основе столь отличающихся технологий, нужны *дополнительные средства*, и такие средства предоставляет сетевой уровень.

Функции сетевого уровня реализуются:

- группой протоколов;
- специальными устройствами — маршрутизаторами.

Одной из функций маршрутизатора является *физическое соединение сетей*. Маршрутизатор имеет несколько сетевых интерфейсов, подобных интерфейсам компьютера, к каждому из которых может быть подключена одна сеть. Таким образом, все интерфейсы маршрутизатора можно считать узлами разных сетей. Маршрутизатор может быть реализован программно на базе универсального компьютера (например, типовая конфигурация Unix или Windows включает программный модуль маршрутизатора). Однако чаще маршрутизаторы реализуются на базе специализированных аппаратных платформ. В состав программного обеспечения маршрутизатора входят протокольные модули сетевого уровня.

Итак, чтобы связать сети, показанные на рис. 4.8, необходимо соединить все эти сети маршрутизаторами и установить протокольные модули сетевого уровня на все конечные узлы пользователей, которые хотели бы связываться через составную сеть (рис. 4.9).

Данные, которые необходимо передать через составную сеть, поступают на сетевой уровень от вышележащего транспортного уровня. Эти данные снабжаются заголовком сетевого уровня. Данные вместе с заголовком образуют **пакет** — так называется PDU сетевого уровня. Заголовок пакета сетевого уровня имеет унифицированный формат, не зависящий

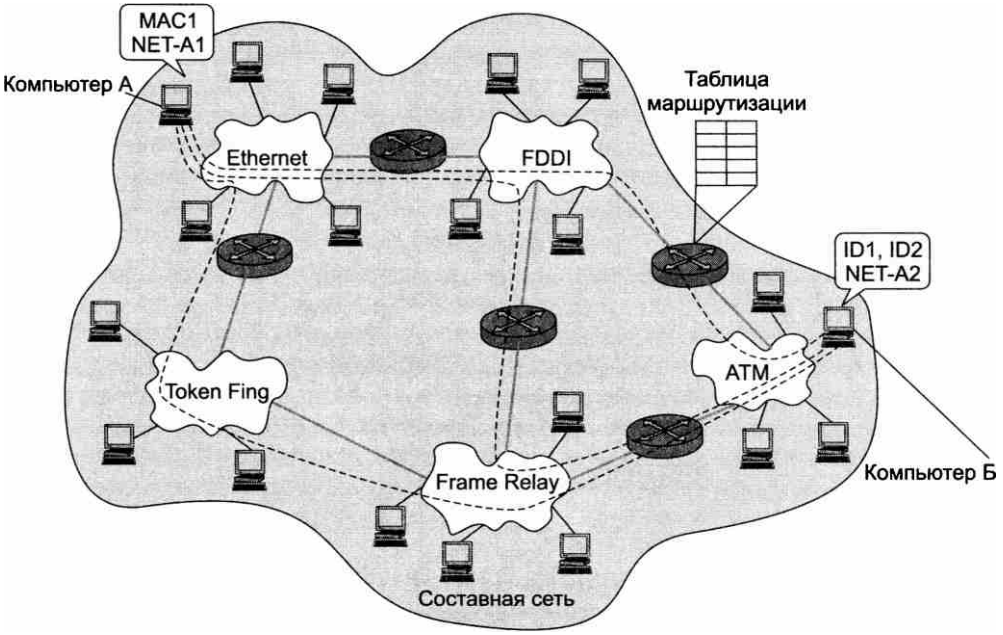


Рис. 4.9. Пример составной сети

от форматов кадров канального уровня тех сетей, которые могут входить в составную сеть, и наряду с другой служебной информацией несет данные об адресе назначения этого пакета.

Для того чтобы протоколы сетевого уровня могли доставлять пакеты любому узлу составной сети, эти узлы должны иметь адреса, уникальные в пределах данной составной сети. Такие адреса называются **сетевыми**, или **глобальными**. Каждый узел составной сети, который намерен обмениваться данными с другими узлами составной сети, наряду с адресом, назначенным ему на канальном уровне, должен иметь сетевой адрес. Например, на рис. 4.9 компьютер в сети Ethernet, входящей в составную сеть, имеет адрес канального уровня MAC1 и адрес сетевого уровня NET-A1; аналогично в сети ATM узел, адресуемый идентификаторами виртуальных каналов ID1 и ID2, имеет сетевой адрес NET-A2. В пакете в качестве адреса назначения должен быть указан адрес сетевого уровня, на основании которого определяется маршрут пакета. *Определение маршрута* является важной задачей сетевого уровня. Маршрут описывается последовательностью сетей (или маршрутизаторов), через которые должен пройти пакет, чтобы попасть к адресату. Например, на рис. 4.9 штриховой линией показаны три маршрута, по которым могут быть переданы данные от компьютера *А* к компьютеру *Б*. Маршрутизатор собирает информацию о топологии связей между сетями и на основе этой информации строит таблицы коммутации, которые в данном случае носят специальное название **таблиц маршрутизации**. Задачу выбора маршрута мы уже коротко обсуждали в разделе «Обобщенная задача коммутации» главы 2.

В соответствии с многоуровневым подходом сетевой уровень для решения своей задачи обращается к нижележащему канальному уровню. Весь путь через составную сеть раз-

бивается на участки от одного маршрутизатора до другого, причем каждый участок соответствует пути через отдельную сеть.

Для того чтобы передать пакет через очередную сеть, сетевой уровень помещает его в поле данных кадра соответствующей канальной технологии, указывая в заголовке кадра канальный адрес интерфейса следующего маршрутизатора. Сеть, используя свою канальную технологию, доставляет кадр с инкапсулированным в него пакетом по заданному адресу. Маршрутизатор извлекает пакет из прибывшего кадра и после необходимой обработки передает пакет для дальнейшей транспортировки в следующую сеть, предварительно упаковав его в новый кадр канального уровня в общем случае другой технологии. Таким образом, сетевой уровень играет роль координатора, организующего совместную работу сетей, построенных на основе разных технологий.

Есть еще одна причина существования сетевого уровня помимо сглаживания различий технологий канального уровня. Сетевой уровень позволяет разбить большую сеть на подсети и управлять каждой из подсетей независимо. Составная сеть с иерархической двухуровневой структурой «канальный уровень» — «сетевой уровень» оказывается гораздо более масштабируемой, чем сеть с одноуровневой структурой, что и показала успешная история Интернета. Даже в условиях доминирования одной технологии канального уровня Ethernet построение всемирной сети с единой одноуровневой структурой оказалось практически невозможным. Поэтому сегодня Интернет представляет собой большое количество локальных и глобальных сетей Ethernet, объединенных общим сетевым уровнем, на котором работает протокол IP. Ну и нельзя исключать ситуацию, что в будущем появятся новые технологии канального уровня и функция их согласования сетевым уровнем снова станет востребована.

Пример-аналогия

Можно найти аналогию между функционированием сетевого уровня и международной почтовой службы, такой, например, как DHL или TNT (рис. 4.10). Представим, что некоторый

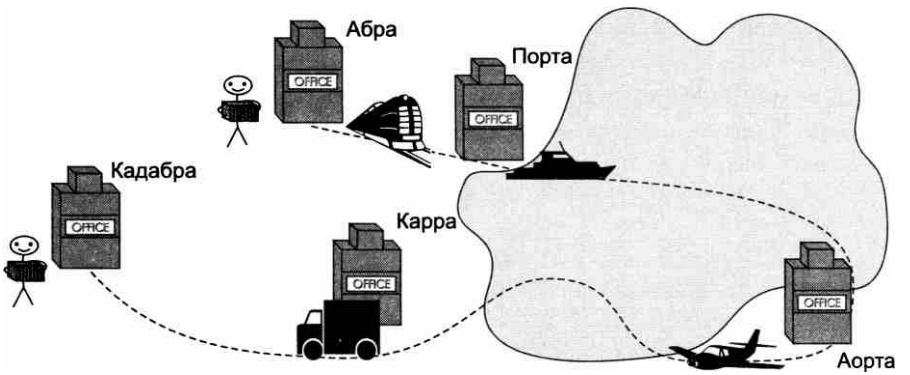


Рис. 4.10. Работа международной почтовой службы

груз необходимо доставить из города Абра в город Кадабра, причем эти города расположены на разных континентах. Для доставки груза международная почта использует услуги различных региональных перевозчиков:

- железную дорогу;
- морской транспорт;
- авиаперевозчиков;
- автомобильный транспорт.

Эти перевозчики могут рассматриваться как аналоги сетей канального уровня, причем каждая «сеть» здесь построена на основе собственной технологии. Из этих региональных служб международная почтовая служба должна организовать единую слаженно работающую сеть. Для этого международная почтовая служба должна, во-первых, продумать маршрут перемещения почты, во-вторых, координировать работу в пунктах смены перевозчиков (например, выгружать почту из вагонов и размещать ее в транспортных отсеках самолетов). Каждый же перевозчик ответствен только за перемещение почты по своей части пути и не несет никакой ответственности за состояние почты за его пределами.

В общем случае функции сетевого уровня шире, чем обеспечение обмена в пределах составной сети. Так, сетевой уровень решает задачу создания надежных и гибких барьеров на пути нежелательного трафика между сетями.

В заключение отметим, что на сетевом уровне определяются два вида протоколов. Первый вид — **маршрутизируемые протоколы** — реализуют продвижение пакетов через сеть. Именно эти протоколы обычно имеют в виду, когда говорят о протоколах сетевого уровня. Однако часто к сетевому уровню относят и другой вид протоколов, называемых **маршрутизирующими протоколами**, или **протоколами маршрутизации**. С помощью этих протоколов маршрутизаторы собирают информацию о топологии межсетевых соединений, на основании которой осуществляется выбор маршрута продвижения пакетов.

Транспортный уровень

На пути от отправителя к получателю пакеты могут быть искажены или утеряны. Хотя некоторые приложения имеют собственные средства обработки ошибок, существуют и такие, которые предпочитают сразу иметь дело с надежным соединением.

Транспортный уровень (transport layer) обеспечивает приложениям и верхним уровням стека — прикладному, представления и сеансовому — передачу данных с той степенью надежности, которая им требуется. Модель OSI определяет пять **классов транспортного сервиса**: от низшего класса 0 до высшего класса 4. Эти виды сервиса отличаются качеством предоставляемых услуг: срочностью, возможностью восстановления прерванной связи, наличием средств мультиплексирования нескольких соединений между различными прикладными протоколами через общий транспортный протокол, а главное — способностью к обнаружению и исправлению ошибок передачи, таких как искажение, потеря и дублирование пакетов.

Выбор класса сервиса транспортного уровня определяется, с одной стороны, тем, в какой степени задача обеспечения надежности решается самими приложениями и протоколами более высоких, чем транспортный, уровней. С другой стороны, этот выбор зависит от того, насколько надежной является система транспортировки данных в сети, обеспечиваемая уровнями, расположенными ниже транспортного: сетевым, канальным и физическим. Так, если качество каналов передачи очень высокое и вероятность возникновения ошибок, не обнаруженных протоколами более низких уровней, невелика, то разумно воспользоваться одним из облегченных сервисов транспортного уровня, не обремененных многочисленными проверками, квитированием и другими приемами повышения надежности. Если же

транспортные средства нижних уровней очень ненадежны, то целесообразно обратиться к наиболее развитому сервису транспортного уровня, который работает, используя максимум средств для обнаружения и устранения ошибок, включая предварительное установление логического соединения, контроль доставки сообщений по контрольным суммам и циклической нумерации пакетов, установление тайм-аутов доставки и т. п.

Все протоколы, начиная с транспортного уровня и выше, реализуются программными средствами конечных узлов сети — компонентами их сетевых операционных систем. В качестве примера транспортных протоколов можно привести протоколы TCP и UDP стека TCP/IP и протокол SPX стека Novell.

Протоколы нижних четырех уровней обобщенно называют сетевым транспортом, или транспортной подсистемой, так как они полностью решают задачу транспортировки сообщений с заданным уровнем качества в составных сетях с произвольной топологией и различными технологиями. Оставшиеся три верхних уровня решают задачи предоставления прикладных сервисов, используя нижележащую транспортную подсистему.

Сеансовый уровень

Сеансовый уровень (session layer) управляет взаимодействием сторон: фиксирует, какая из сторон является активной в настоящий момент, и предоставляет средства синхронизации сеанса. Эти средства позволяют в ходе длинных передач сохранять информацию о состоянии этих передач в виде контрольных точек, чтобы в случае отказа можно было вернуться назад к последней контрольной точке, а не начинать все сначала. На практике немногие приложения используют сеансовый уровень, и он редко реализуется в виде отдельных протоколов. Функции этого уровня часто объединяют с функциями прикладного уровня и реализуют в одном протоколе.

Уровень представления

Уровень представления (presentation layer), как явствует из его названия, обеспечивает представление передаваемой по сети информации, не меняя при этом ее содержания. За счет уровня представления информация, передаваемая прикладным уровнем одной системы, всегда понятна прикладному уровню другой системы. С помощью средств данного уровня протоколы прикладных уровней могут преодолеть синтаксические различия в представлении данных или же различия в кодах символов, например кодов ASCII и EBCDIC. На этом уровне могут выполняться шифрование и дешифрирование данных, благодаря которым секретность обмена данными обеспечивается сразу для всех прикладных служб. Примером такого протокола является протокол SSL (Secure Socket Layer — слой защищенных сокетов), который обеспечивает секретный обмен сообщениями для протоколов прикладного уровня стека TCP/IP.

Прикладной уровень

Прикладной уровень (application layer) — это в действительности просто набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к общим ресурсам, таким как файлы, принтеры или гипертекстовые веб-страницы, а также организуют

свою совместную работу, например по протоколу электронной почты. Единица данных, которой оперирует прикладной уровень, обычно называется **сообщением**.

Существует очень большое разнообразие протоколов и соответствующих служб прикладного уровня. Приведем в качестве примера несколько наиболее распространенных протоколов этого уровня:

- протоколы доступа к файлам NFS и FTP в стеке TCP/IP, SMB в Microsoft Windows, NCP в операционной системе Novell NetWare;
- почтовые протоколы SMTP, IMAP, POP3;
- протокол передачи гипертекстовых сообщений HTTP.

Модель OSI и сети с коммутацией каналов

Как уже было упомянуто, модель OSI описывает процесс взаимодействия устройств в сети с **коммутацией пакетов**. А как же обстоит дело с сетями **коммутации каналов**? Существует ли для них собственная справочная модель? Можно ли сопоставить функции технологий коммутации каналов с уровнями модели OSI?

Да, для представления структуры средств межсетевого взаимодействия сетей с коммутацией каналов также используется многоуровневый подход, в соответствии с которым существуют протоколы нескольких уровней, образующих иерархию. Однако общей справочной модели, подобной модели OSI, для сетей с коммутацией каналов не существует. Например, различные типы телефонных сетей имеют собственные стеки протоколов, отличающиеся количеством уровней и распределением функций между уровнями. Первичные сети, такие как SDH или DWDM, также обладают собственной иерархией протоколов. Ситуация усложняется еще и тем, что практически все типы современных сетей с коммутацией каналов задействуют эту технику только для передачи пользовательских данных, а для управления процессом установления соединений в сети и общего управления сетью применяют технику коммутации пакетов. Такими сетями являются, например, сети ISDN, SDH, DWDM.

Для сетей с коммутацией пакетов сети с коммутацией каналов предоставляют сервис физического уровня, хотя сами они устроены достаточно сложно и поддерживают собственную иерархию протоколов.

Рассмотрим, к примеру, случай, когда несколько локальных пакетных сетей связываются между собой через цифровую телефонную сеть. Очевидно, что функции создания составной сети выполняют протоколы сетевого уровня, поэтому мы устанавливаем в каждой локальной сети маршрутизатор. Маршрутизатор должен быть оснащен интерфейсом, способным установить соединение через телефонную сеть с другой локальной сетью. После того как такое соединение установлено, в телефонной сети образуется поток битов, передаваемых с постоянной скоростью. Это соединение и предоставляет маршрутизаторам сервис физического уровня. Для того чтобы организовать передачу данных, маршрутизаторы используют поверх этого физического канала какой-либо двухточечный протокол канального уровня.

Стандартизация сетей

Универсальный тезис о пользе стандартизации, справедливый для всех отраслей, в компьютерных сетях приобретает особое значение. Суть сети — это соединение разного обо-

рудования, а значит, проблема совместимости является здесь одной из наиболее острых. Без согласования всеми производителями общепринятых стандартов для оборудования и протоколов прогресс в деле «строительства» сетей был бы невозможен. Поэтому все развитие компьютерной отрасли в конечном счете отражено в стандартах — любая новая технология только тогда приобретает «законный» статус, когда ее содержание закрепляется в соответствующем стандарте.

В компьютерных сетях идеологической основой стандартизации является рассмотренная ранее модель взаимодействия открытых систем (OSI).

Понятие открытой системы

Что же такое открытая система?

Открытой может быть названа любая система (компьютер, вычислительная сеть, ОС, программный пакет, другие аппаратные и программные продукты), которая построена в соответствии с открытыми спецификациями.

Напомним, что под термином «спецификация» в вычислительной технике понимают формализованное описание аппаратных или программных компонентов, способов их функционирования, взаимодействия с другими компонентами, условий эксплуатации, особых характеристик. Понятно, что не всякая спецификация является стандартом.

Под **открытыми спецификациями** понимаются опубликованные, общедоступные спецификации, соответствующие стандартам и принятые в результате достижения согласия после всестороннего обсуждения всеми заинтересованными сторонами.

Использование при разработке систем открытых спецификаций позволяет третьим сторонам создавать для этих систем различные аппаратные или программные средства расширения и модификации, а также программно-аппаратные комплексы из продуктов разных производителей.

Открытый характер стандартов и спецификаций важен не только для коммуникационных протоколов, но и для разнообразных устройств и программ, выпускаемых для построения сети. Нужно отметить, что большинство стандартов, принимаемых сегодня, носят открытый характер. Время закрытых систем, точные спецификации на которые были известны только фирме-производителю, ушло. Все осознали, что возможность взаимодействия с продуктами конкурентов не снижает, а, наоборот, повышает ценность изделия, так как позволяет применять его в большем количестве работающих сетей, собранных из продуктов разных производителей. Поэтому даже такие фирмы, как IBM и Microsoft, ранее выпускавшие закрытые системы, сегодня активно участвуют в разработке открытых стандартов и применяют их в своих продуктах.

Для реальных систем полная открытость является недостижимым идеалом. Как правило, даже в системах, называемых открытыми, этому определению соответствуют лишь некоторые части, поддерживающие внешние интерфейсы. Например, открытость семейства операционных систем Unix заключается помимо всего прочего в наличии стандартизованного программного интерфейса между ядром и приложениями, что позволяет легко переносить приложения из среды одной версии Unix в среду другой версии.

Модель OSI касается только одного аспекта открытости, а именно открытости средств взаимодействия устройств, связанных в компьютерную сеть. Здесь под открытой системой понимается сетевое устройство, готовое взаимодействовать с другими сетевыми

устройствами по стандартным правилам, определяющим формат, содержание и значение принимаемых и отправляемых сообщений.

Если две сети построены с соблюдением принципов открытости, это дает следующие преимущества:

- возможность построения сети из аппаратных и программных средств различных производителей, придерживающихся одного и того же стандарта;
- безболезненная замена отдельных компонентов сети другими, более совершенными, что позволяет сети развиваться с минимальными затратами;
- легкость сопряжения одной сети с другой.

Источники стандартов

Закон РФ № 65-ФЗ «О техническом регулировании» определяет понятие «стандарт» следующим образом:

> Стандарт — это «документ, в котором в целях добровольного многократного использования устанавливаются характеристики продукции, правила осуществления и характеристики процессов проектирования (включая изыскания), производства, строительства, монтажа, наладки, эксплуатации, хранения, перевозки, реализации и утилизации, выполнения работ или оказания услуг. Стандарт также может содержать правила и методы исследований (испытаний) и измерений, правила отбора образцов, требования к терминологии, символике, упаковке, маркировке или этикеткам и правилам их нанесения».

По умолчанию *соблюдение стандарта не является обязательным* (если явно не указана обязательность его исполнения). Однако существует множество причин, по которым большинство компаний, предприятий, частных лиц, организаций добровольно выбирают следование стандартам. Мы уже говорили о стандартизации как средстве обеспечения совместимости информационных технологий, продуктов и терминологии. Следование стандартам позволяет также создавать более качественные, более конкурентоспособные технологии, системы и услуги, так как стандарты — это концентрированное выражение передовой технической мысли, они аккумулируют актуальные теоретические знания и так называемые «лучшие практики».

В Законе РФ о техническом регулировании говорится, что разработчиком стандарта может быть любое лицо, но, как правило, стандарты разрабатываются рабочими группами (техническими комитетами), в состав которых на добровольной основе могут включаться представители органов исполнительной власти, научных, коммерческих и некоммерческих организаций, общественных объединений. Одним из основных принципов стандартизации является ориентация на тех лиц, кто в наибольшей степени заинтересован в существовании стандартов. Поэтому очень часто разработчиками стандартов являются компании и организации, много и успешно работающие в той области, для которой они предлагают стандарты.

В зависимости от статуса организаций различают следующие виды стандартов:

- *стандарты отдельных фирм*, например стек протоколов SNA компании IBM или графический интерфейс OPEN LOOK для Unix-систем компании Sun;

- *стандарты специальных комитетов и объединений* создаются несколькими компаниями, например стандарты технологии ATM, разрабатываемые специально созданным объединением ATM Forum, которое насчитывает около 100 коллективных участников, или стандарты союза Fast Ethernet Alliance, касающиеся технологии 100 Мбит Ethernet;
- *национальные стандарты*, например стандарт FDDI, представляющий один из многочисленных стандартов института ANSI, или стандарты безопасности для операционных систем, разработанные центром NCSC министерства обороны США;
- *международные стандарты*, например модель и стек коммуникационных протоколов Международной организации по стандартизации (ISO), многочисленные стандарты Международного союза электросвязи (ITU), в том числе стандарты на сети с коммутацией пакетов X.25, сети Frame Relay, ISDN, модемы и многие другие.

В нашей стране главную организационную роль в стандартизации играет Федеральное агентство по техническому регулированию и метрологии (Росстандарт). Росстандарт создает и координирует рабочие группы по разработке стандартов, организует общественное обсуждение и экспертизу новых стандартов, утверждает и публикует документы по стандартам, ведет учет и распространение национальных стандартов.

Некоторые стандарты, непрерывно развиваясь, могут переходить из одной категории в другую. В частности, фирменные стандарты на продукцию, получившую широкое распространение, обычно становятся международными стандартами де-факто, так как вынуждают производителей из разных стран следовать фирменным стандартам, чтобы обеспечить совместимость своих изделий с этими популярными продуктами. Например, из-за феноменального успеха персонального компьютера компании IBM фирменный стандарт на архитектуру IBM PC стал международным стандартом де-факто.

Более того, ввиду широкого распространения некоторые фирменные стандарты становятся основой для национальных и международных стандартов де-юре. Например, стандарт Ethernet, первоначально разработанный компаниями Digital Equipment, Intel и Xerox, через некоторое время и в несколько измененном виде был принят как национальный стандарт IEEE 802.3, а затем организация ISO утвердила его в качестве международного стандарта ISO 8802.3.

Стандартизация Интернета

Ярким примером открытой системы является Интернет. Эта международная сеть развивалась в полном соответствии с требованиями, предъявляемыми к открытым системам. В разработке ее стандартов принимали участие тысячи специалистов — пользователей этой сети из различных университетов, научных организаций и фирм — производителей вычислительной аппаратуры и программного обеспечения, работающих в разных странах. Само название стандартов, определяющих работу Интернета, — **темы для обсуждения** (Request For Comments, RFC) — показывает гласный и открытый характер принимаемых стандартов. В результате Интернет сумел объединить в себе разнообразное оборудование и программное обеспечение огромного числа сетей, разбросанных по всему миру.

Ввиду постоянно растущей популярности Интернета RFC-документы становятся международными стандартами де-факто, многие из которых затем приобретают статус офици-

альных международных стандартов в результате их утверждения какой-либо организаций по стандартизации, как правило, ISO и ITU-T.

Существует несколько организационных подразделений, отвечающих за развитие и, в частности, за стандартизацию архитектуры и протоколов Интернета. Основным из них является научно-административное **сообщество Интернета** (Internet Society, ISOC), объединяющее около 100 000 человек, которое занимается социальными, политическими и техническими проблемами эволюции Интернета.

Под управлением ISOC работает **совет по архитектуре Интернета** (Internet Architecture Board, IAB). В IAB входят две основные группы: Internet Research Task Force (IRTF) и Internet Engineering Task Force (IETF). IRTF координирует долгосрочные исследовательские проекты по протоколам TCP/IP. IETF — это инженерная группа, которая занимается решением текущих технических проблем Интернета. Именно IETF определяет спецификации, которые затем становятся стандартами Интернета. Процесс разработки и принятия стандарта для протокола Интернета состоит из ряда обязательных этапов, или стадий, включающих непременную экспериментальную проверку.

В соответствии с принципом открытости Интернета все RFC-документы в отличие, скажем, от стандартов ISO находятся в свободном доступе. Список RFC-документов можно найти, в частности, на сайте www.rfc-editor.org.

Стандартные стеки коммуникационных протоколов

Важнейшим направлением стандартизации в области вычислительных сетей является стандартизация коммуникационных протоколов. Наиболее известными стеками протоколов являются: OSI, TCP/IP, IPX/SPX, NetBIOS/SMB, DECnet, SNA (не все из них применяются сегодня на практике).

Стек OSI

Важно различать *модель* OSI и *стек протоколов* OSI. В то время как модель OSI является концептуальной схемой взаимодействия открытых систем, стек OSI представляет собой набор спецификаций конкретных протоколов.

В отличие от других стеков протоколов стек OSI полностью соответствует модели OSI, включая спецификации протоколов для всех семи уровней взаимодействия, определенных в этой модели (рис. 4.11). Это и понятно, разработчики стека OSI использовали модель OSI как прямое руководство к действию.

Протоколы стека OSI отличают сложность и неоднозначность спецификаций. Эти свойства стали результатом общей политики разработчиков стека, стремившихся учесть в своих протоколах все многообразие уже существующих и появляющихся технологий.

На *физическом* и *канальном уровнях* стек OSI поддерживает протоколы Ethernet, Token Ring, FDDI, а также протоколы LLC, X.25 и ISDN, то есть, как и большинство других стеков, использует все разработанные вне стека популярные протоколы нижних уровней.

Сетевой уровень включает сравнительно редко используемые протоколы Connection-oriented Network Protocol (CONP) и Connectionless Network Protocol (CLNP). Как следует из названий, первый из них ориентирован на соединение (connection-oriented), второй — нет (connectionless).

Рис. 4.11. Стек протоколов OSI

Более популярны протоколы маршрутизации стека OSI: между конечной и промежуточной системами (End System — Intermediate System, ES-IS) и между промежуточными системами (Intermediate System — Intermediate System, IS-IS).

Транспортный уровень стека OSI в соответствии с функциями, определенными для него в модели OSI, скрывает различия между сетевыми сервисами с установлением соединения и без установления соединения, так что пользователи получают требуемое качество обслуживания независимо от нижележащего сетевого уровня. Чтобы обеспечить это, транспортный уровень требует, чтобы пользователь задал нужное качество обслуживания.

Службы *прикладного уровня* обеспечивают передачу файлов, эмуляцию терминала, сервис каталогов и почту. Из них наиболее популярными являются сервис каталогов (стандарт X.500), электронная почта (X.400), протокол виртуального терминала (VTP), протокол передачи, доступа и управления файлами (FTAM), протокол пересылки и управления работами (JTM).

Стек NetBIOS/SMB

Стек NetBIOS/SMB является совместной разработкой компаний IBM и Microsoft (рис. 4.12). На физическом и канальном уровнях этого стека также задействованы уже получившие распространение протоколы, такие как Ethernet, Token Ring, FDDI, а на верхних уровнях — специфические протоколы NetBEUI и SMB.

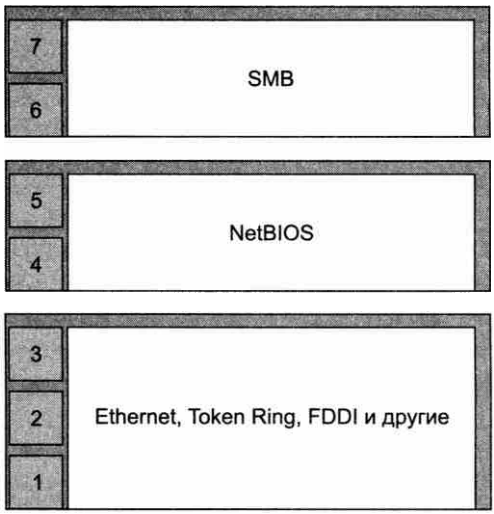


Рис. 4.12. Стек NetBIOS/SMB

Протокол Network Basic Input/Output System (NetBIOS) появился в 1984 году как сетевое расширение стандартных функций базовой системы ввода-вывода (BIOS) IBM PC для сетевой программы PC Network фирмы IBM. В дальнейшем этот протокол был заменен так называемым протоколом расширенного пользовательского интерфейса NetBEUI (NetBIOS Extended User Interface). Для совместимости приложений в качестве интерфейса к протоколу NetBEUI был сохранен интерфейс NetBIOS. NetBEUI разрабатывался как эффективный протокол, потребляющий немного ресурсов и предназначенный для сетей, насчитывающих не более 200 рабочих станций. Этот протокол поддерживает много полезных сетевых функций, которые можно отнести к транспортному и сеансовому уровням модели OSI, однако с его помощью *невозможна маршрутизация* пакетов. Это ограничивает применение протокола NetBEUI локальными сетями, не разделенными на подсети, и делает невозможным его использование в составных сетях.

Протокол Server Message Block (SMB) поддерживает функции сеансового уровня, уровня представления и прикладного уровня. На основе SMB реализуется файловая служба, а также службы печати и передачи сообщений между приложениями.

Стек TCP/IP

Стек TCP/IP был разработан по инициативе Министерства обороны США более 20 лет назад для связи экспериментальной сети ARPAnet с другими сетями как набор общих протоколов для разнородной вычислительной среды. Большой вклад в развитие стека TCP/IP, который получил свое название по популярным протоколам IP и TCP, внес университет Беркли, реализовав протоколы стека в своей версии ОС UNIX. Популярность этой операционной системы привела к широкому распространению протоколов TCP, IP и других протоколов стека. Сегодня этот стек используется для связи компьютеров в Интернете,

а также в огромном числе корпоративных сетей. Мы подробно рассмотрим этот стек протоколов в части IV, посвященной сетям TCP/IP.

Соответствие популярных стеков протоколов модели OSI

На рис. 4.13 показано, в какой степени популярные стеки протоколов соответствуют рекомендациям модели OSI. Как мы видим, часто это соответствие весьма условно. В большинстве случаев разработчики стеков отдавали предпочтение скорости работы сети в ущерб модульности — ни один стек, кроме стека OSI, не разбит на семь уровней. Чаще всего в стеке явно выделяются 3–4 уровня: уровень сетевых адаптеров, в котором реализуются протоколы физического и канального уровней, сетевой уровень, транспортный уровень и уровень служб, вбирающий в себя функции сеансового уровня, уровня представления и прикладного уровня.

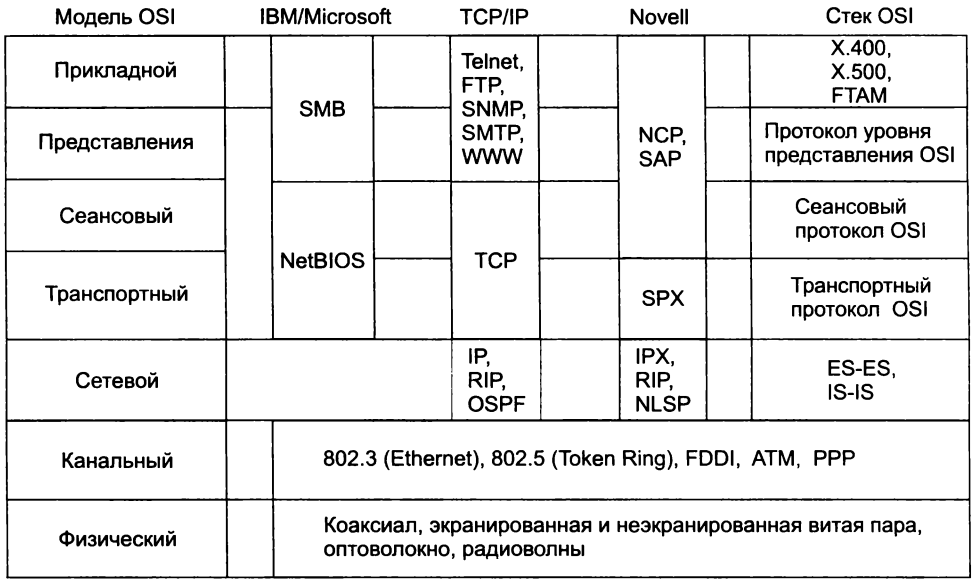


Рис. 4.13. Соответствие популярных стеков протоколов модели OSI

Структура стеков протоколов часто не соответствует рекомендуемому моделью OSI разбиению на уровни и по другим причинам. Давайте вспомним, чем характеризуется идеальная многоуровневая декомпозиция. С одной стороны, необходимо соблюсти принцип иерархии: каждый вышележащий уровень обращается с запросами только к нижележащему, а нижележащий предоставляет свои сервисы только непосредственно соседствующему с ним вышележащему. В стеках протоколов это приводит к тому, что PDU вышележащего уровня всегда инкапсулируется в PDU нижележащего.

С другой же стороны, идеальная многоуровневая декомпозиция предполагает, что все модули, отнесенные к одному уровню, ответственны за решение общей для всех них

задачи. Однако эти требования часто вступают в противоречие. Например, основной функцией протоколов сетевого уровня стека TCP/IP (так же, как и сетевого уровня OSI) является передача пакетов через составную сеть. Для решения этой задачи в стеке TCP/IP предусмотрено несколько протоколов: протокол продвижения IP-пакетов, протоколы маршрутизации RIP, OSPF и др. Если считать признаком принадлежности к одному и тому же уровню общность решаемых задач, то, очевидно, протокол IP и протоколы маршрутизации должны быть отнесены к одному уровню. Вместе с тем если принять во внимание, что сообщения протокола RIP инкапсулируются в UDP-дейтаграммы, а сообщения протокола OSPF — в IP-пакеты, то, следуя формально принципу иерархической организации стека, OSPF следовало бы отнести к транспортному, а RIP — к прикладному уровню. На практике же протоколы маршрутизации обычно включают в сетевой уровень.

Информационные и транспортные услуги

Услуги компьютерной сети можно разделить на две категории:

- транспортные услуги;
- информационные услуги.

Транспортные услуги состоят в передаче информации между пользователями сети в неизменном виде. При этом сеть принимает информацию от пользователя на одном из своих интерфейсов, передает ее через промежуточные коммутаторы и выдает другому пользователю через другой интерфейс. При оказании транспортных услуг сеть не вносит никаких изменений в передаваемую информацию, передавая ее получателю в том виде, в котором она поступила в сеть от отправителя. Примером транспортной услуги глобальных сетей является объединение локальных сетей клиентов.

Информационные услуги состоят в предоставлении пользователю некоторой новой информации. Информационная услуга всегда связана с операциями по обработке информации: хранению ее в некотором упорядоченном виде (файловая система, база данных, веб-сайт), поиску нужной информации и преобразованию информации. Информационные услуги существовали и до появления первых компьютерных сетей, например справочные услуги телефонной сети. С появлением компьютеров информационные услуги пережили революцию, так как компьютер и был изобретен для автоматической программной обработки информации. Для оказания информационных услуг применяются различные информационные технологии: программирование, управление базами данных и файловыми архивами, веб-сервис, электронная почта.

В телекоммуникационных сетях «докомпьютерной» эры всегда преобладали транспортные услуги. Основной услугой телефонной сети была передача голосового трафика между абонентами, в то время как справочные услуги были дополнительными. В компьютерных сетях одинаково важны обе категории услуг. Эта особенность компьютерных сетей сегодня отражается на названии нового поколения телекоммуникационных сетей, которые появляются в результате конвергенции сетей различных типов. Такие сети все чаще стали называть **инфокоммуникационными**. Это название пока не стало общеупотребительным, но оно хорошо отражает новые тенденции, включая обе составляющие услуг на равных правах.

Распределение протоколов по элементам сети

На рис. 4.14 показаны основные элементы компьютерной сети: конечные узлы — компьютеры и промежуточные узлы — коммутаторы и маршрутизаторы.

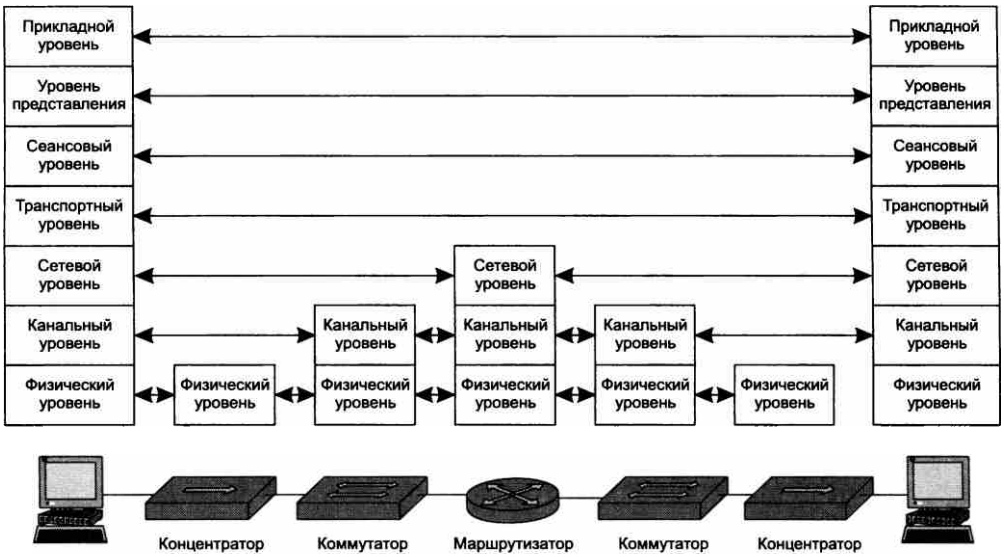


Рис. 4.14. Соответствие функций различных устройств сети уровням модели OSI

Из рисунка видно, что полный стек протоколов реализован только на конечных узлах, а промежуточные узлы поддерживают протоколы всего трех нижних уровней. Это объясняется тем, что коммуникационным устройствам для продвижения пакетов достаточно функциональности нижних трех уровней. Более того, коммуникационное устройство может поддерживать только протоколы двух нижних уровней или даже одного физического уровня — это зависит от типа устройства.

Именно к таким устройствам, работающим на физическом уровне, относятся, например, сетевые повторители, называемые также концентраторами, или хабами. Они повторяют электрические сигналы, поступающие на одни их интерфейсы, на других своих интерфейсах, улучшая их характеристики — мощность и форму сигналов, синхронность их следования.

Коммутаторы локальных сетей поддерживают протоколы двух нижних уровней, физического и канального, что дает им возможность работать в пределах стандартных топологий.

Маршрутизаторы должны поддерживать протоколы всех трех уровней, так как сетевой уровень нужен им для объединения сетей различных технологий, а протоколы нижних уровней — для взаимодействия с конкретными сетями, образующими составную сеть, например Ethernet или Frame Relay.

Коммутаторы глобальных сетей (например, MPLS), работающие на основе технологии виртуальных каналов, могут поддерживать как два уровня протоколов, так и три. Протокол сетевого уровня нужен им в том случае, если они поддерживают процедуры автоматического

установления виртуальных каналов. Так как топология глобальных сетей произвольная, без сетевого протокола обойтись нельзя. Если же виртуальные соединения устанавливаются администраторами сети вручную, то коммутатору глобальной сети достаточно поддерживать только протоколы физического и канального уровней, чтобы передавать данные по уже проложенным виртуальным каналам.

Компьютеры, на которых работают сетевые приложения, должны поддерживать протоколы всех уровней. Протоколы прикладного уровня, пользуясь сервисами протоколов уровня представления и сеансового уровня, предоставляют приложениям набор сетевых услуг в виде сетевого прикладного программного интерфейса (API). Протокол транспортного уровня также работает на всех конечных узлах. При передаче данных через сеть два модуля транспортного протокола, работающие на узле-отправителе и узле-получателе, взаимодействуют друг с другом для поддержания транспортного сервиса нужного качества. Коммуникационные устройства сети переносят сообщения транспортного протокола прозрачным образом, не вникая в их содержание.

В компьютерах коммуникационные протоколы всех уровней (кроме физического и части функций канального уровня) реализуются программно операционной системой или системными приложениями.

Конечные узлы сети (компьютеры и компьютеризованные устройства, например мобильные телефоны) всегда предоставляют как информационные, так и транспортные услуги, а промежуточные узлы сети — только транспортные. Когда мы говорим, что некоторая сеть предоставляет *только транспортные услуги*, то подразумеваем, что конечные узлы находятся за границей сети. Это обычно имеет место в обслуживающих клиентов коммерческих сетях.

Если же говорят, что сеть предоставляет *также информационные услуги*, то это значит, что компьютеры, предоставляющие эти услуги, включаются в состав сети. Примером является типичная ситуация, когда поставщик услуг Интернета поддерживает еще и собственные веб-серверы.

Вспомогательные протоколы транспортной системы

Настало время сказать, что на рис. 4.14 показан упрощенный вариант распределения протоколов между элементами сети. В реальных сетях некоторые из коммуникационных устройств поддерживают не только протоколы трех нижних уровней, но и протоколы верхних уровней. Так, маршрутизаторы реализуют протоколы маршрутизации, позволяющие автоматически строить таблицы маршрутизации, а концентраторы и коммутаторы часто поддерживают протоколы SNMP и telnet, которые не нужны для выполнения основных функций этих устройств, но позволяют конфигурировать их и управлять ими удаленно. Существуют также DNS-серверы, которые отображают символьные имена хостов на их IP-адреса, и без этой вспомогательной функции нормальная работа в Интернете практически невозможна.

Большинство вспомогательных протоколов формально относится к прикладному уровню модели OSI, так как в своей работе они обращаются к протоколам нижних уровней, таким как TCP, UDP или SSL. Однако при этом вспомогательные протоколы не переносят

пользовательские данные, то есть они не выполняют непосредственно функций протокола прикладного уровня, описанного в модели OSI.

Очевидно, что при рассмотрении вспомогательных протоколов мы сталкиваемся с ситуацией, когда деления протоколов на уровни иерархии (то есть деления «по вертикали»), присущего модели OSI, оказывается недостаточно. Полезным оказывается деление протоколов на группы «по горизонтали».

При горизонтальном делении все протоколы (как основные, так и вспомогательные) разделяют на три слоя (рис. 4.15):

- **пользовательский слой** (user plane) включает группу основных протоколов, то есть протоколов, которые переносят пользовательский трафик;
- **слой управления** (control plane) составляют вспомогательные протоколы, необходимые для работы основных протоколов сети, например протоколы маршрутизации, протоколы отображения имен на IP-адреса;
- **слой менеджмента** (management plane) объединяет вспомогательные протоколы, поддерживающие операции менеджмента (управления сетью администратором), такие как протокол SNMP для сбора информации об ошибках, протоколы удаленного конфигурирования устройств.

Рис. 4.15. Три группы протоколов

«Горизонтальное» деление протоколов снимает сложности, возникающие при соотнесении некоторых протоколов уровням модели OSI. Например, в книгах одних авторов протоколы маршрутизации могут находиться на сетевом уровне, в книгах других — на прикладном. Это происходит не из-за небрежности авторов, а из-за объективных трудностей классификации. Модель OSI хорошо подходит для стандартизации протоколов, которые переносят пользовательский трафик, то есть протоколов пользовательского слоя. В то же время она в гораздо меньшей степени годится для определения места вспомогательных протоколов в общей модели функционирования сети. Поэтому многим авторам приходится помещать протоколы маршрутизации на сетевой уровень, чтобы таким образом отразить функциональную близость этих протоколов к операции продвижения пакетов.

Классификация компьютерных сетей

Классификация — процесс группирования (отнесения к тому или иному типу) объектов изучения в соответствии с их общими признаками.

Каждый реальный объект может быть наделен множеством признаков. *Субъективный* характер любой классификации проявляется в том, что имеется некоторый произвол при выборе среди этого множества признаков тех, которые будут использованы для классификации, то есть при выборе **критериев классификации**. Приведенная далее классификация компьютерных сетей не является исключением — в других источниках (да и далее в этой книге) вы можете встретить другие признаки, по которым сети относят к тому или иному типу.

Начнем с того, что компьютерные сети сами собой являются элементом классификации телекоммуникационных сетей, а именно телекоммуникационные сети по виду передаваемого контента делятся на:

- радиосети;
- телефонные сети;
- телевизионные сети;
- компьютерные сети.

В зависимости от *территории покрытия* компьютерные сети можно разделить на три группы:

- **локальные сети** (Local Area Network, LAN);
- **глобальные сети** (Wide Area Network, WAN);
- **городские сети**, или **сети мегаполиса** (Metropolitan Area Network, MAN).

Подчеркнем, что, говоря в данном контексте «локальные сети» или «глобальные сети», мы имеем в виду прежде всего различия *технологий* локальных и глобальных сетей, а не тот факт, что эти сети имеют разный территориальный масштаб.

Мы уже обозначили особенности двух этих направлений, когда рассматривали в главе 1 эволюцию компьютерных сетей. В частности, в локальных сетях качество линий связи между узлами обычно выше, чем в глобальных сетях. Это обусловлено различными причинами:

- существенно меньшей длиной линий связи (метры вместо сотен километров), а значит, и меньшими искажениями сигналов, вносимых неидеальной передающей средой;
- меньшим уровнем внешних помех, так как в локальной сети оборудование и кабели обычно размещаются в специальных защищенных экранированных помещениях, а линии связи глобальной сети могут проходить в сильно электромагнитно «зашумленной» среде, например в туннелях подземных коммуникаций, рядом с силовыми кабелями, вдоль линий электропередач и т. п.;
- экономическими соображениями.

Высокое качество линий связи и низкий уровень помех позволили упростить процедуры передачи данных в технологиях локальных сетей, например применять простые методы

кодирования и модулирования сигналов, отказаться от сложных алгоритмов восстановления искаженных данных.

Несмотря на то что процесс сближения технологий локальных и глобальных сетей идет уже давно, различия между этими технологиями все еще достаточно отчетливы, что и дает основания относить соответствующие сети к различным технологическим типам.

Сети MAN предназначены для обслуживания территории крупного города — мегаполиса, и сочетают в себе признаки как локальных, так и глобальных сетей. От первых они унаследовали большую плотность подключения конечных абонентов и высокоскоростные линии связи, а от последних — большую протяженность линий связи.

В соответствии с технологическими признаками, обусловленными *средой передачи*, компьютерные сети подразделяют на два класса:

- **проводные сети** — сети, каналы связи которых построены с использованием медных или оптических кабелей;
- **беспроводные сети** — сети, в которых для связи используются беспроводные каналы связи, например радио, СВЧ, инфракрасные или лазерные каналы.

Любая беспроводная среда — будь то радиоволны, инфракрасные лучи или СВЧ-сигналы спутниковой связи — гораздо больше подвержена влиянию внешних помех, чем проводная. Роса, туман, солнечные бури, работающие в комнате микроволновые печи — вот только несколько примеров источников помех, которые могут привести к резкому ухудшению качества беспроводного канала. А значит, технологии беспроводных сетей должны учитывать типичность таких ситуаций и строиться таким образом, чтобы обеспечивать работоспособность сети, несмотря на ухудшение вешних условий. Здесь, как и в случае локальных и глобальных сетей, мы сталкиваемся с влиянием качества линий связи на технологию передачи данных.

Кроме того, существует ряд других специфических особенностей беспроводных сетей, которые служат основанием для выделения их в особый класс, например естественное разделение радиосреды всеми узлами сети, находящимися в радиусе действия всенаправленного передатчика; распределение диапазона радиочастот между сетями различного назначения, например между телефонными и компьютерными.

В зависимости от способа *коммутации* сети подразделяются на два фундаментально различных класса:

- **сети с коммутацией пакетов**;
- **сети с коммутацией каналов**.

Сейчас в компьютерных сетях преимущественно используется техника коммутации пакетов, хотя принципиально допустимо и применение в них техники коммутации каналов.

Техника коммутации пакетов, в свою очередь, допускает несколько вариаций, отличающихся способом продвижения пакетов, в соответствии с чем сети делятся на:

- **дейтаграммные сети**, например Ethernet;
- **сети, основанные на логических соединениях**, например IP-сети, использующие на транспортном уровне протокол TCP;
- **сети, основанные на виртуальных каналах**, например MPLS-сети.

Сети могут быть классифицированы на основе *топологии*. Топологический тип сети весьма отчетливо характеризует сеть, он понятен как профессионалам, так и пользователям. Мы

подробно рассматривали базовые топологии сетей, поэтому здесь только перечислим их: **полносвязная топология**, **дерево**, **звезда**, **кольцо**, **смешанная топология**.

В зависимости от того, *какому типу пользователей предназначаются услуги сети*, сети делятся на сети операторов связи, корпоративные и персональные сети.

- **Сети операторов связи** предоставляют публичные услуги, то есть клиентом сети может стать любой индивидуальный пользователь или любая организация, которая заключила соответствующий коммерческий договор на предоставление той или иной телекоммуникационной услуги. Традиционными услугами операторов связи являются услуги телефонии, а также предоставления каналов связи в аренду тем организациям, которые собираются строить на их основе собственные сети. С распространением компьютерных сетей операторы связи существенно расширили спектр своих услуг, добавив к ним услуги Интернета, услуги виртуальных частных сетей, веб-хостинг, электронную почту и IP-телефонию, а также широковещательную рассылку аудио- и видеосигналов. Интернет сегодня также является одной из разновидностей сети операторов связи. Интернет быстро превратился из сети, обслуживающей сравнительно немногочисленное академическое сообщество, во всемирную публичную сеть, предоставляющую набор наиболее востребованных услуг для всех.
- **Корпоративные сети** предоставляют услуги только сотрудникам предприятия, которое владеет этой сетью. Хотя формально корпоративная сеть может иметь любой размер, обычно под корпоративной понимают сеть крупного предприятия, которая состоит как из локальных сетей, так и из объединяющей их глобальной сети.
- **Персональные сети** находятся в личном использовании. Для них характерно небольшое количество узлов, простая структура, а также небольшой (в пределах 30 метров) радиус действия. Узлами персональной сети наряду с настольными компьютерами могут быть телефоны, смартфоны, планшеты, ноутбуки. Чаще всего персональные сети строятся на основе беспроводных технологий.

В зависимости от *функциональной роли*, которую играют некоторые части сети, ее относят к сети доступа, магистральной сети или сети агрегирования трафика (рис. 4.16).

- **Сети доступа** — это сети, предоставляющие доступ индивидуальным и корпоративным абонентам от их помещений (квартир, офисов) до первого помещения (пункта присутствия) оператора сети связи или оператора корпоративной сети. Другими словами, это сети, ответственные за расширение глобальной сети до помещений ее клиентов.
- **Магистральные сети** — это сети, представляющие собой наиболее скоростную часть (ядро) глобальной сети, которая объединяет многочисленные сети доступа в единую сеть.
- **Сети агрегирования трафика** — это сети, агрегирующие данные от многочисленных сетей доступа для компактной передачи их по небольшому числу каналов связи в магистраль. Сети агрегирования обычно используются только в крупных глобальных сетях, где они занимают промежуточную позицию, помогая магистральной сети обрабатывать трафик, поступающий от большого числа сетей доступа. В сетях среднего и небольшого размера сети агрегирования обычно отсутствуют.

Различают также первичные и наложенные телекоммуникационные сети:

- **Первичные сети** занимают особое положение в мире телекоммуникационных сетей, их можно рассматривать как *вспомогательные* сети, позволяющие гибко создавать

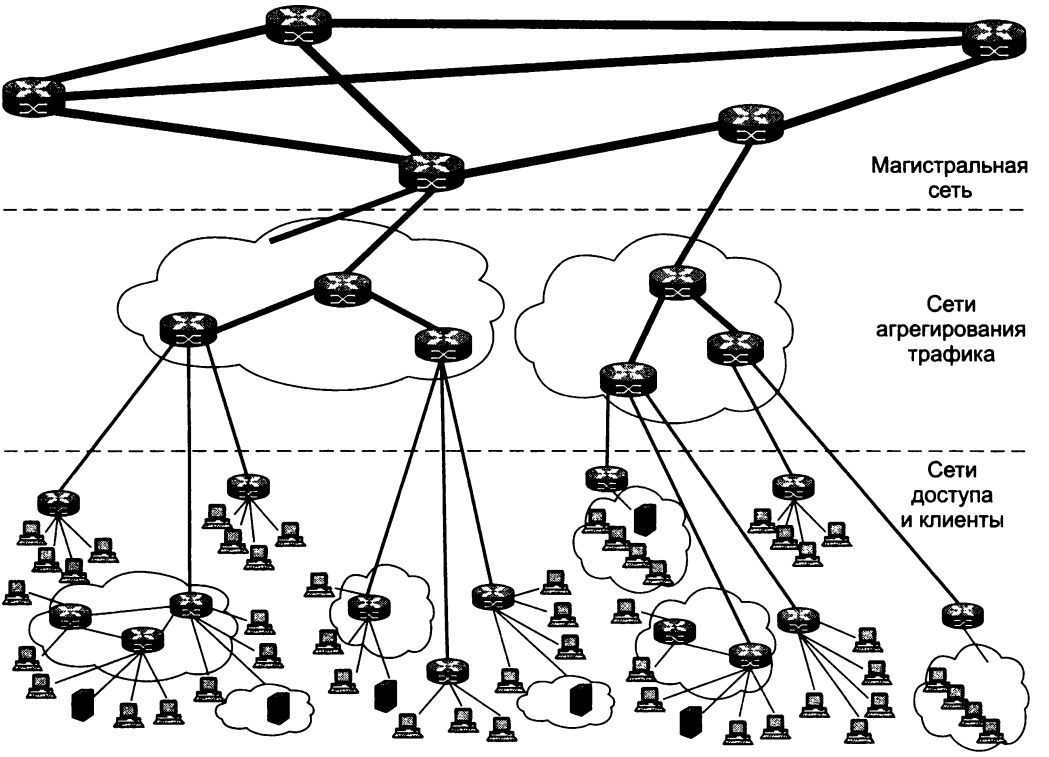


Рис. 4.16. Сети доступа, сети агрегирования трафика и магистральная сеть

постоянные физические двухточечные каналы для других компьютерных и телефонных сетей. В соответствии с семиуровневой моделью OSI первичные сети подобно простым кабелям выполняют функции физического уровня сетей. Однако в отличие от кабелей первичные сети включают дополнительное коммуникационное оборудование, которое путем соответствующего конфигурирования позволяет прокладывать новые физические каналы между конечными точками сети. Другими словами, первичная сеть — это гибкая среда для создания физических каналов связи.

- **Наложенные сети** в этой классификации — это все остальные сети, которые предоставляют услуги конечным пользователям и строятся на основе каналов первичных сетей — «накладываются» поверх этих сетей. То есть и компьютерные, и телефонные, и телевизионные сети являются наложенными.

Оптоволоконные кабели обладают наилучшими на сегодняшний день характеристиками передачи данных, они используются как в локальных, так и в глобальных проводных сетях. Тем не менее термин **оптические сети** часто трактуется специалистами по компьютерным сетям и телекоммуникациям в узком смысле: как синоним первичных сетей. Это объясняется тем, что передача данных по оптическим кабелям является для первичных сетей основным вариантом работы, а использование других сред передачи не может обеспечить в первичных сетях высоких современных требований по скорости и надежности обмена информацией между удаленными узлами сети.

Интернет представляет собой уникальную сеть, объединяющую практически все компьютерные сети (за исключением, может быть, сетей, остающихся изолированными по причине повышенной секретности) во всемирном масштабе. Если применить к Интернету признаки, описанные в классификации, можно сказать, что это:

- публичная сеть;
- сеть операторов связи, предоставляющая публичные услуги, как информационные, так и транспортные;
- сеть с коммутацией пакетов;
- сеть, состоящая из магистральных сетей, сетей агрегирования трафика и сетей доступа.

Выводы

Эффективной моделью средств взаимодействия компьютеров в сети является многоуровневая структура, в которой модули вышележащего уровня при решении своих задач рассматривают средства нижележащего уровня как некий инструмент. Каждый уровень данной структуры поддерживает интерфейсы двух типов. Во-первых, это интерфейсы услуг с выше- и нижележащими уровнями «своей» иерархии средств. Во-вторых, это одноранговый интерфейс со средствами другой взаимодействующей стороны, расположенными на том же уровне иерархии. Этот интерфейс называют протоколом.

Иерархически организованный набор протоколов, достаточный для взаимодействия узлов в сети, называется стеком протоколов. Протоколы нижних уровней часто реализуются комбинацией программных и аппаратных средств, а протоколы верхних уровней — как правило, программными средствами. Программный модуль, реализующий некоторый протокол, называют протокольной сущностью, или тоже протоколом.

В начале 80-х годов ISO, ITU-T при участии некоторых других международных организаций по стандартизации разработали стандартную модель взаимодействия открытых систем (OSI). Модель OSI содержит описание обобщенного представления средств сетевого взаимодействия и используется в качестве своего рода универсального языка сетевых специалистов, именно поэтому ее называют справочной моделью. Модель OSI определяет 7 уровней взаимодействия, дает им стандартные имена, указывает, какие функции должен выполнять каждый уровень.

Открытой системой может быть названа любая система (компьютер, компьютерная сеть, операционная система, программный пакет, другие аппаратные и программные продукты), которая построена в соответствии с общедоступными спецификациями и стандартами, принятыми в результате публичного обсуждения всеми заинтересованными сторонами.

В зависимости от области действия различают стандарты отдельных компаний, стандарты специальных комитетов и объединений, национальные стандарты, международные стандарты.

Важнейшим направлением стандартизации в области вычислительных сетей является стандартизация коммуникационных протоколов. Примерами стандартизованных стеков протоколов являются TCP/IP, IPX/SPX, NetBIOS/SMB, OSI, DECnet, SNA. Лидирующее положение занимает стек TCP/IP, он используется для связи десятков миллионов компьютеров всемирной информационной сети Интернет. Стек TCP/IP имеет 4 уровня:

прикладной, транспортный, межсетевого взаимодействия и сетевых интерфейсов. Соответствие уровней стека TCP/IP уровням модели OSI достаточно условно.

Компьютерные сети предоставляют услуги двух типов: информационные и транспортные. Информационные услуги предоставляются конечными узлами сети — серверами, а транспортные — промежуточными узлами, которыми являются коммутаторы и маршрутизаторы сети.

Классификация компьютерных сетей может быть выполнена на основе различных критериев. Это могут быть технологические характеристики сетей, такие как топология, метод коммутации, метод продвижения пакетов, тип используемой среды передачи. Сети классифицируют и на основе других признаков, не являющихся технологическими, таких, например, как тип потребителей предоставляемых услуг (сети операторов и корпоративные сети) или функциональная роль (магистраль, сеть доступа).

Контрольные вопросы

1. Можно ли представить еще один вариант модели взаимодействия открытых систем с другим количеством уровней, например 8 или 5?
2. Какие из приведенных утверждений не всегда справедливы? Варианты ответов:
   а) протокол — это стандарт, описывающий правила взаимодействия двух систем;
   б) протокол — это формализованное описание правил взаимодействия, включая последовательность обмена сообщениями и их форматы;
   в) логический интерфейс — это формализованное описание правил взаимодействия, включая последовательность обмена сообщениями и их форматы.
3. Пусть на двух компьютерах установлено идентичное программное и аппаратное обеспечение, за исключением того, что драйверы сетевых адаптеров Ethernet поддерживают разные интерфейсы с протоколом сетевого уровня IP. Будут ли эти компьютеры нормально взаимодействовать, если их соединить в сеть?
4. Какое минимальное количество уровней протоколов (в терминах модели OSI) должны поддерживать маршрутизаторы сетей с коммутацией пакетов?
5. К какому типу сети относится Интернет?

ГЛАВА 5 Сетевые характеристики

Типы характеристик

Компьютерная сеть представляет собой сложную и дорогую систему, решающую ответственные задачи и обслуживающую большое количество пользователей. Поэтому очень важно, чтобы сеть не просто работала, но работала качественно.

Понятие *качества обслуживания* можно трактовать очень широко, включая в него все возможные и желательные для пользователя свойства сети и поставщика услуг, поддерживающего работу этой сети. Для того чтобы пользователь и поставщик услуг могли более конкретно обсуждать проблемы обслуживания и строить свои отношения на формальной основе, существует ряд общепринятых характеристик качества предоставляемых сетью услуг. Мы будем рассматривать в этой главе только характеристики качества *транспортных* услуг сети, которые намного проще поддаются формализации, чем характеристики качества информационных услуг. Характеристики качества транспортных услуг отражают такие важнейшие свойства сети, как производительность, надежность и безопасность.

Часть этих характеристик может быть оценена количественно и измерена при обслуживании пользователя. Пользователь и поставщик услуг могут заключить соглашение об уровне обслуживания, в котором оговорить требования к количественным значениям некоторых характеристик, например к доступности предоставляемых услуг.

Субъективные оценки качества

Если опросить пользователей, чтобы выяснить, что они вкладывают в понятие качественных сетевых услуг, то можно получить очень широкий спектр ответов. Среди них, скорее всего, встретятся следующие мнения:

- сеть работает быстро, без задержек;
- трафик передается надежно, данные не теряются;
- услуги предоставляются бесперебойно по схеме 24×7 (то есть 24 часа в сутки семь дней в неделю);
- служба поддержки работает хорошо, давая полезные советы и помогая разрешить проблемы;
- услуги предоставляются по гибкой схеме, мне нравится, что можно в любой момент и в широких пределах повысить скорость доступа к сети и увеличить число точек доступа;
- поставщик не только передает мой трафик, но и защищает мою сеть от вирусов и атак злоумышленников;
- я всегда могу проконтролировать, насколько быстро и без потерь сеть передает мой трафик;

- поставщик предоставляет широкий спектр услуг, в частности помимо стандартного доступа в Интернет он предлагает хостинг для моего персонального веб-сайта и услуги IP-телефонии.

Эти *субъективные* оценки отражают *пожелания пользователей* к качеству сетевых сервисов.

Количественные характеристики и требования

Пользователи сети — это хотя и важная, но только одна сторона бизнеса сетей передачи данных. Существует и другая сторона — *поставщик услуг*, коммерческий, если это публичная сеть, или некоммерческий, если сеть корпоративная. Для того чтобы обе стороны — пользователи и поставщики услуг — могли «найти общий язык», существуют *формализованные количественные характеристики* качества сетевых услуг.

Получая сетевые услуги, пользователь формулирует определенные *требования к характеристикам сети*. Например, пользователь может потребовать, чтобы средняя скорость передачи его информации через сеть не опускалась ниже 2 Мбит/с. То есть в данном случае пользователь задает тот диапазон значений для средней скорости передачи информации через сеть, который для него означает хорошее качество сервиса.

Все множество характеристик качества транспортных услуг сети можно отнести к одной из следующих групп:

- производительность;
- надежность;
- безопасность;
- характеристики, имеющие значение только для поставщика услуг.

Первые три группы соответствуют трем наиболее важным для пользователя характеристикам транспортных услуг — возможности без потерь и перерывов в обслуживании (**надежность**) передавать с заданной скоростью (**производительность**) защищенную от несанкционированного доступа и подмены информацию (**безопасность**). Понятно, что поставщик сетевых услуг, стремясь удовлетворить требования клиентов, также уделяет внимание этим характеристикам. В то же время существует ряд характеристик, важных для поставщика сети, но не представляющих интереса для пользователей.

Дело в том, что сеть обслуживает большое количество клиентов и поставщику услуг нужно организовать работу своей сети таким образом, чтобы одновременно удовлетворить требования *всех* пользователей. Как правило, это сложная проблема, так как основные ресурсы сети — линии связи и коммутаторы (маршрутизаторы) — разделяются между информационными потоками пользователей. Поставщику необходимо найти такой баланс в распределении ресурсов между конкурирующими потоками, чтобы требования всех пользователей были соблюдены. Решение этой задачи включает *планирование* и *контроль* расходования ресурсов в процессе передачи пользовательского трафика. Например, его интересует производительность коммутатора, так как поставщик должен оценить, какое количество потоков пользователей он может обработать с помощью данного коммутатора. Для пользователя же производительность коммутатора никакого значения не имеет, ему важен конечный результат — будет его поток обслужен качественно или нет.

Временна́я шкала

Рассмотрим еще один способ классификации характеристик — в соответствии с временно́й шкалой, на которой эти характеристики определяются.

Долговременные характеристики (или **характеристики проектных решений**) определяются на промежутках времени от нескольких месяцев до нескольких лет. Примерами таких характеристик являются количество и схема соединения коммутаторов в сети, пропускная способность линий связи, конкретные модели и характеристики используемого оборудования. Эти параметры сети прямо влияют на характеристики качества услуг сети. Одно проектное решение может оказаться удачным и сбалансированным, так что потоки трафика не будут испытывать перегрузок; другое может создавать узкие места для потоков, в результате задержки и потери пакетов превысят допустимые пределы. Понятно, что полная замена или глубокая модернизация сети связана с большими финансовыми и временны́ми затратами, поэтому они не могут происходить часто, а значит, выбранные однажды параметры продолжают оказывать влияние на качество функционирования сети в течение продолжительного времени.

Среднесрочные характеристики определяются на интервалах времени от нескольких секунд до нескольких дней. Как правило, за это время происходит обслуживание большого количества пакетов. Например, к среднесрочным характеристикам может быть отнесено усредненное значение задержки пакетов по выборке, взятой в течение суток.

Краткосрочные характеристики относятся к темпу обработки отдельных пакетов и измеряются в микросекундном и миллисекундном диапазонах. Например, время буферизации, или время пребывания пакета в очереди коммутатора либо маршрутизатора, является характеристикой этой группы. Для анализа и обеспечения требуемого уровня краткосрочных характеристик разработано большое количество методов, получивших название методов **контроля и предотвращения перегрузок**.

Соглашение об уровне обслуживания

Основой нормального сотрудничества поставщика услуг и пользователей является *договор*. Такой договор заключается всегда, однако далеко не всегда в нем указываются количественные требования к эффективности предоставляемых услуг. Очень часто в договоре услуга специфицируется очень общо, например «предоставление доступа в Интернет».

Однако существует и другой тип договора, называемый **соглашением об уровне обслуживания** (Service Level Agreement, SLA). В таком соглашении поставщик услуг и клиент описывают качество предоставляемой услуги в количественных терминах, пользуясь характеристиками эффективности сети. Например, в SLA может быть записано, что поставщик обязан передавать трафик клиента без потерь и с той средней скоростью, с которой пользователь направляет его в сеть. При этом оговорено, что это соглашение действует только в том случае, если средняя скорость трафика пользователя не превышает, например, 3 Мбит/с, в противном случае поставщик получает право просто не передавать избыточный трафик. Для того чтобы каждая сторона могла контролировать соблюдение этого соглашения, необходимо еще указать период времени, на котором будет измеряться средняя скорость, например день, час или секунда. Еще более определенным соглашение об уровне обслуживания становится в том случае, когда в нем указываются средства и методы измерения характеристик сети, чтобы у поставщика и пользователя не было расхождений при контроле соглашения.

Соглашения об уровне обслуживания могут заключаться не только между поставщиками коммерческих услуг и их клиентами, но и между подразделениями одного и того же предприятия. В этом случае поставщиком сетевых услуг может являться, например, отдел информационных технологий, а потребителем — производственный отдел.

Производительность

Мы уже знакомы с такими важными долговременными характеристиками производительности сетевого оборудования, как пропускная способность каналов или производительность коммутаторов и маршрутизаторов. Наибольший интерес данные характеристики представляют для поставщиков услуг — на их основе поставщик услуг может планировать свой бизнес, рассчитывая максимальное количество клиентов, которое он может обслужить, определяя рациональные маршруты прохождения трафика и т. п.

Однако клиента интересуют другие характеристики производительности, которые позволят ему количественно оценить, насколько быстро и качественно сеть передает его трафик. Для того чтобы определить эти характеристики, воспользуемся моделью идеальной сети.

Идеальная сеть

В разделе «Количественное сравнение задержек» главы 3 мы рассмотрели различные составляющие задержек в сети с коммутацией пакетов. Напомним, что такими составляющими являются показатели времени:

- передачи данных в канал (время сериализации);
- распространения сигнала;
- ожидания пакета в очереди;
- коммутации пакета.

Две первые составляющие задержки полностью определяются свойствами каналов передачи данных (битовой скоростью и скоростью распространения сигнала в среде) и являются фиксированными для пакета фиксированной длины.

Две последние составляющие зависят от характеристик сети коммутации пакетов (загрузки коммутаторов и их быстродействия) и для пакета фиксированной длины в общем случае являются переменными.

Будем считать, что сеть с коммутацией пакетов работает идеально, если она передает каждый бит информации с постоянной скоростью, равной скорости распространения света в используемой физической среде. Другими словами, **идеальная сеть с коммутацией пакетов** не вносит никаких дополнительных задержек в передачу данных помимо тех, которые вносятся каналами связи, то есть две последние составляющие задержки равны нулю.

Результат передачи пакетов такой идеальной сетью иллюстрирует рис. 5.1. На верхней оси показаны значения времени поступления пакетов в **сеть** от узла отправителя, а на нижней — значения времени поступления пакетов в узел назначения. Говорят, что верхняя ось показывает **предложенную нагрузку** сети, а нижняя — результат передачи этой нагрузки через сеть.

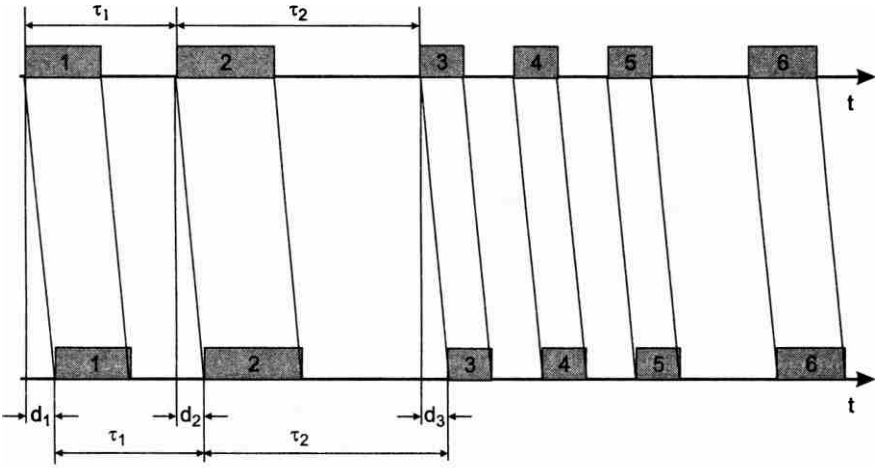


Рис. 5.1. Передача пакетов идеальной сетью

Пусть задержка передачи пакета определяется как интервал времени между моментом отправления первого бита пакета в канал связи узлом отправления и моментом поступления первого бита пакета в узел назначения соответственно (на рисунке обозначены задержки d_1, d_2 и d_3 пакетов 1, 2 и 3 соответственно).

Как видно из рисунка, идеальная сеть доставляет все пакеты узлу назначения:

- не потеряв ни одного из них (и не исказив информацию ни в одном из них);
- в том порядке, в котором они были отправлены;
- с одной и той же и минимально возможной задержкой ($d_1 = d_2$ и т. д.).

Важно, что все интервалы между соседними пакетами сеть сохраняет в неизменном виде. Например, если интервал между первым и вторым пакетами составляет при отправлении τ_1 секунд, а между вторым и третьим — τ_2, то такими же интервалы останутся в узле назначения.

Надежная доставка всех пакетов с минимально возможной задержкой и сохранением временны́х интервалов между ними удовлетворит любого пользователя сети независимо от того, трафик какого приложения он передает по сети — веб-сервиса или IP-телефонии.

Существуют и другие определения времени задержки пакета. Например, эту величину можно определить как время между моментом отправления первого бита пакета в канал связи узлом отправления и моментом поступления последнего бита пакета в узел назначения соответственно (такое определение используется в документе RFC 2679, описывающем характеристики задержек IP-пакетов). Нетрудно видеть, что в этом определении в задержку пакета включено время сериализации, кроме того, понятно, что оба определения не противоречат друг другу и величина задержки, полученная в соответствии с одним определением, легко преобразуется в величину задержки, полученной в соответствии с другим. Мы выбрали первое определение для иллюстрации идеального поведения сети с коммутацией пакетов потому, что в этом случае задержка не зависит от размера пакета, что проще использовать, описывая «идеальность» обслуживания пакетов.

Теперь посмотрим, какие отклонения от идеала могут встречаться в *реальной* сети и какими характеристиками можно эти отклонения описывать (рис. 5.2).

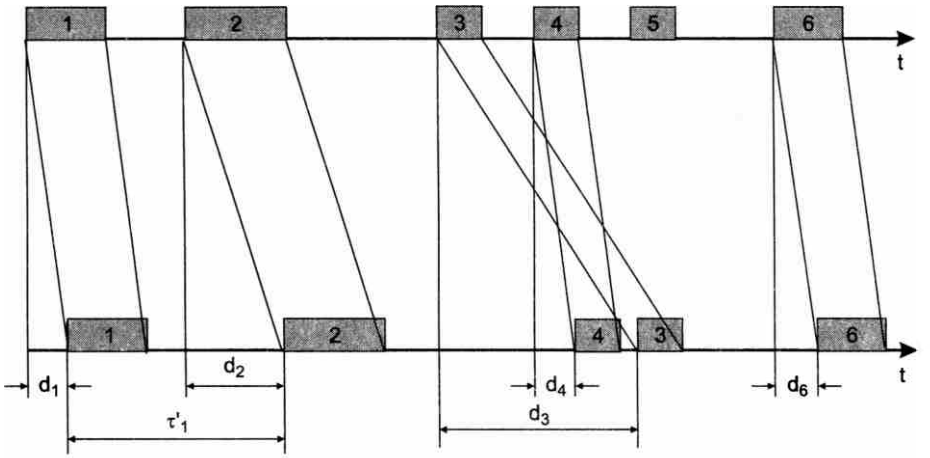


Рис. 5.2. Передача пакетов реальной сетью

Пакеты доставляются сетью узлу назначения *с различными задержками*. Как мы уже знаем, это неотъемлемое свойство сетей с коммутацией пакетов.

Случайный характер процесса образования очередей приводит к случайным задержкам, при этом задержки отдельных пакетов могут быть значительными, в десятки раз превосходя среднюю величину задержек ($d_1 \neq d_2 \neq d_3$ и т. д.). Неравномерность задержек изменяет относительное положение пакетов в выходном потоке, а это может катастрофически сказаться на качестве работы некоторых приложений. Например, при цифровой передаче речи исходный поток представляет собой равномерно отстоящие друг от друга пакеты, несущие замеры голоса. Неравномерность интервалов между пакетами выходного потока приводит к существенным искажениям речи.

Пакеты могут доставляться узлу назначения *не в том порядке*, в котором они были отправлены, например на рис. 5.2 пакет 4 поступил в узел назначения раньше, чем пакет 3. Такие ситуации встречаются в дейтаграммных сетях, когда различные пакеты одного потока передаются через сеть различными маршрутами, а следовательно, ожидают обслуживания в разных очередях с разным уровнем задержек. Очевидно, что пакет 3 проходил через перегруженный узел или узлы, так что его суммарная задержка оказалась настолько большой, что пакет 4 прибыл раньше него.

Пакеты *могут теряться* в сети или же приходить в узел назначения *с искаженными данными*, что равносильно потере пакета, так как большинство протоколов не способно восстанавливать искаженные данные, а только определяет этот факт по значению контрольной суммы в заголовке кадра.

Пакеты также могут *дублироваться* по разным причинам, например из-за ошибочных повторных передач пакета, предпринятых протоколом, в котором таким образом обеспечивается надежный обмен данными.

В реальной сети средняя скорость информационного потока на входе узла назначения может отличаться от средней скорости потока, направленного в сеть узлом-отправителем. Виной этому являются не задержки пакетов, а их потери[1]. Так, в примере, показанном на рис. 5.2, *средняя скорость исходящего потока снижается* из-за потери пакета 5. Чем больше потерь и искажений пакетов происходит в сети, тем ниже скорость информационного потока.

Как видно из приведенного описания, существуют различные **характеристики производительности сети** (называемые также **метриками производительности сети**). Нельзя в общем случае говорить, что одни их этих характеристик более, а другие — менее важные. Относительная важность характеристик зависит от типа приложения, трафик которого переносит сеть. Так, существуют приложения, которые очень чувствительны к задержкам пакетов, но в то же время весьма терпимы к потере отдельного пакета — примером может служить передача голоса через пакетную сеть. Примером приложения, которое, напротив, мало чувствительно к задержкам пакетов, но очень чувствительно к их потерям, является загрузка файлов (подробнее об этом говорится в следующей главе). Поэтому для каждого конкретного случая необходимо выбирать подходящий набор характеристик сети, адекватно отражающий влияние «неидеальности» сети на работу приложения.

Статистические оценки характеристик сети

Для оценки характеристик *случайных процессов* служат статистические методы, а именно такой характер имеют процессы передачи пакетов сетью. Сами характеристики производительности сети, такие как, например, задержка пакета, являются *случайными величинами*. Статистические характеристики выявляют закономерности в поведении сети, которые устойчиво проявляются только в длительных периодах времени. Когда мы говорим о длительном периоде времени, то мы понимаем под этим интервал, в миллионы раз больший, чем время передачи одного пакета, которое в современной сети измеряется микросекундами. Так, время передачи пакета Fast Ethernet составляет около 100 мкс, Gigabit Ethernet — около 10 мкс, ячейки ATM — от долей микросекунды до 3 мкс (в зависимости от скорости передачи). Поэтому для получения устойчивых результатов нужно наблюдать поведение сети по крайней мере в течение минут, а лучше — нескольких часов.

Основным инструментом статистики является так называемая **гистограмма** распределения оцениваемой случайной величины. Рассмотрим, например, гистограмму задержки пакета. Будем считать, что нам удалось измерить задержку доставки каждого из 2600 пакетов, переданных между двумя узлами сети, и сохранить полученные результаты. Эти результаты называются **выборкой** случайной величины.

Для того чтобы получить гистограмму распределения, мы должны разбить весь диапазон измеренных значений задержек на несколько интервалов и подсчитать, сколько пакетов из нашей выборки попало в каждый интервал. Пусть все значения задержек укладываются в диапазон 20–90 мс. Разобьем его на семь интервалов по 10 мс. В каждый из этих интервалов, начиная с интервала 20–30 мс и т. д., попало 100 ($n1$), 200 ($n2$), 300 ($n3$), 300 ($n4$), 400 ($n5$), 800 ($n6$) и 500 ($n7$) пакетов соответственно. Отобразив эти числа в виде горизонтальных уровней для каждого интервала, мы получим гистограмму, показанную

[1] Скорость передачи данных определяется как частное от деления объема передаваемых данных на время их передачи (задержку). Из определения следует, что эта характеристика всегда является усредненной.

на рис. 5.3, которая, основываясь всего на семи числах $n1$, $n2$, ..., $n7$, дает нам компактную статистическую характеристику задержек 2600 пакетов.

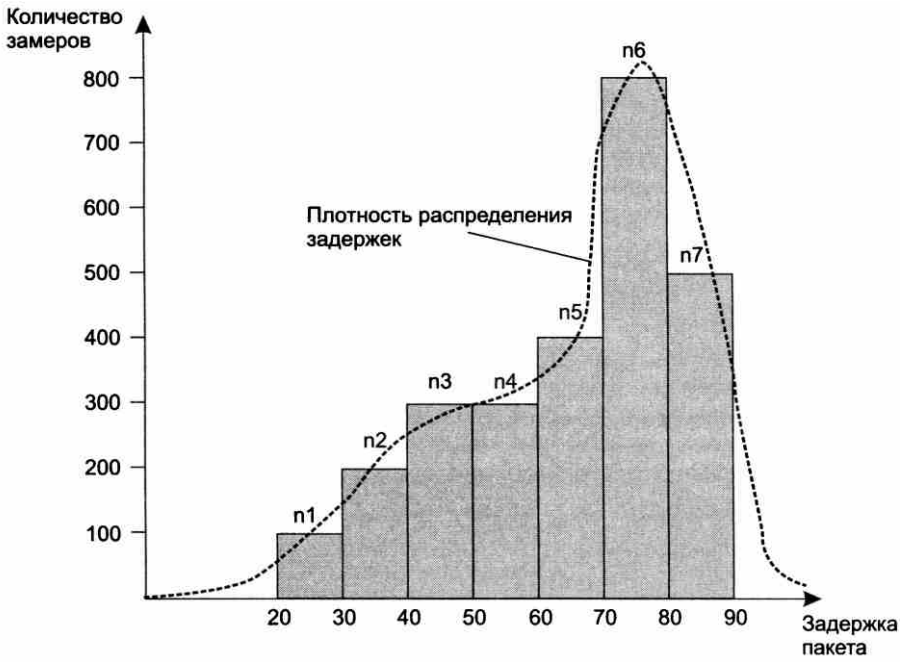


Рис. 5.3. Гистограмма распределения задержек

Гистограмма задержек дает хорошее представление о производительности сети. По ней можно судить, какие уровни задержек более вероятны, а какие — менее. Чем больше период времени, в течение которого собираются данные для построения гистограммы, тем с более высокой степенью достоверности можно предсказать поведение сети в будущем. Например, пользуясь гистограммой на рис. 5.3, можно сказать, что и в будущем при измерениях задержек пакетов у 65 % пакетов задержка превысит 60 мс. Для получения такой оценки мы сложили общее количество пакетов, задержки которых попали во все интервалы, большие 60 мс (1700 замеров), и разделили эту величину на общее количество пакетов (2600 замеров).

Насколько точен такой прогноз? Собрали ли мы достаточно экспериментальных данных, чтобы делать более или менее достоверные прогнозы? Статистика позволяет судить и об этом, однако мы не будем рассматривать здесь эту увлекательную проблему и оставим ее специальным книгам по статистике.

При увеличении количества интервалов и времени наблюдения гистограмма в пределе переходит в непрерывную функцию, которая называется **плотностью распределения** задержки доставки пакета (показана пунктиром). В соответствии с теорией вероятность того, что значение случайной величины окажется в определенном диапазоне, равна интегралу плотности распределения случайной величины от нижней до верхней границы данного диапазона. Таким образом, может быть вычислено вероятностное значение задержки пакета.

Гистограмма дает хорошее графическое описание соответствующей характеристики, но чаще используются более компактные **статистические оценки** характеристик, которые позволяют представить характеристику *одним числом* на основе некоторой математической обработки имеющейся выборки.

Для описания характеристик производительности сети используются следующие статистические оценки.

- **Среднее значение** (D) вычисляется как сумма всех значений оцениваемой величины d_i, деленная на количество всех измерений N:

$$D = \sum \frac{d_i}{N}.$$

  Для примера, приведенного на рис. 5.3, среднее значение равно: $(100 \times 25 + 200 \times 35 + 300 \times 45 + 300 \times 55 + 400 \times 65 + 800 \times 75 + 500 \times 85)/2600 = 64{,}6$ мс (при вычислениях использованы средние значения интервалов).

- **Медиана** представляет такое значение оцениваемой величины, которое делит ранжированную (упорядоченную) выборку пополам, то есть таким образом, чтобы количество замеров, значения которых меньше или равны значению медианы, равнялось количеству замеров, значения которых больше или равны значению медианы. В нашем примере медианой выборки является значение 70 мс, так как число замеров, значения которых меньше или равны 70 мс, составляет 1300, как и число замеров, значения которых больше или равны 70 мс.

- **Стандартное отклонение** (J) представляет собой среднее отклонение каждого отдельного замера от среднего значения оцениваемой величины:

$$J = \sqrt{\frac{\sum (d_i - D)^2}{N - 1}}.$$

  Очевидно, что если все задержки d_i равны между собой, то вариация отсутствует, что подтверждают приведенные формулы, — в этом случае $D = d_i$, и $J = 0$.

- **Коэффициент вариации** (CV) — это безразмерная величина, которая равна отношению стандартного отклонения к среднему значению оцениваемой величины:

$$CV = \frac{J}{D}.$$

  Коэффициент вариации характеризует оцениваемую величину без привязки к ее абсолютным значениям. Так, идеальный равномерный поток пакетов всегда будет обладать нулевым значением коэффициента вариации задержки пакета. Коэффициент вариации задержки пакета, равный 1, означает достаточно пульсирующий трафик, так как средние отклонения интервалов от некоторого среднего периода следования пакетов равны этому периоду.

- **Квантиль (процентиль)** — это значение оцениваемой величины, делящее ранжированную выборку на две части так, что процент замеров, значения которых меньше или равны значению квантиля, равен некоторому заданному уровню. В этом определении фигурируют два числа: заранее заданный процент и найденное по нему и замерам выборки значение квантиля. Рассмотрим для примера выборку задержек пакетов, показанную на рис. 5.3, и найдем для нее значение 50-процентного квантиля. Ответом

будет 70 мс, так как ровно 50 % замеров выборки (то есть 1300 замеров из 2600) имеют значения, меньшие или равные 70 мс. Данные нашего примера позволяют также найти значение 80-процентного квантиля — оно равно 80 мс, так как именно 80 % всех замеров имеют значения задержки менее 80 мс. Нетрудно заметить, что медиана является частным случаем квантиля — это 50-процентный квантиль. Для оценки характеристик сети обычно используют квантили с достаточно большим значением процента, например 90-, 95- или 99-процентные квантили. Это понятно, так как если пользователю скажут, что сеть будет обеспечивать уровень задержек в 100 мс с вероятностью 0,5, то это его не очень обрадует, так как он ничего не будет знать об уровне задержек половины своих пакетов.

Мы рассмотрели применение статистических методов для оценки характеристик производительности сети на примере такой характеристики, как задержка. Естественно, эти методы применяются ко всем другим характеристикам производительности сети — времени ожидания в буфере, времени коммутации, количеству потерянных пакетов и др., так как все они являются случайными величинами.

Активные и пассивные измерения в сети

Для того чтобы оценить некоторую характеристику производительности сети, необходимо провести определенные измерения на последовательности пакетов, поступающих на некоторый интерфейс сетевого устройства. Существуют два типа измерений в сети: активные и пассивные.

Активные измерения основаны на генерации в узле-источнике специальных «измерительных» пакетов. Эти пакеты должны пройти через сеть тот же путь, что и пакеты, характеристики которых мы собираемся оценивать. Измерения в узле назначения проводятся на последовательности «измерительных» пакетов.

Рисунок 5.4 иллюстрирует идею активных измерений. Пусть мы хотим измерить задержки пакетов некоторого приложения *A*, которые передаются от компьютера-клиента приложения *A* компьютеру-серверу приложения *A* через сеть. Вместо того чтобы пытаться измерить задержки пакетов, генерируемых клиентским компьютером, мы устанавливаем в сети два дополнительных компьютера: сервер-генератор и сервер-измеритель. Как это следует из их названий, сервер-генератор генерирует измерительные пакеты (показанные на рисунке серым цветом), а сервер-измеритель измеряет задержки этих пакетов. Для того чтобы измеряемые значения были близки к значениям задержки пакетов приложения *A*, нужно, чтобы измерительные пакеты проходили через сеть по тому же пути, что и пакеты приложения *A*. В нашем примере эта цель достигается за счет подключения дополнительных узлов к портам тех же коммутаторов *S*1 и *S*2, к которым подключены оригинальные узлы. Кроме того, нужно, чтобы измерительные пакеты как можно больше «походили» на оригинальные пакеты — размерами, признаками, помещенными в заголовки пакетов. Это требуется для того, чтобы сеть обслуживала их так же, как оригинальные пакеты.

Измерительные пакеты не должны генерироваться слишком часто, иначе нагрузка сети может существенно измениться и результаты замеров будут отличаться от тех, которые были бы получены в отсутствие измерительных пакетов. Другими словами, измерения не должны менять условий работы сети. Обычно интенсивность генерации измерительных пакетов не превышает 20–50 пакетов в секунду.

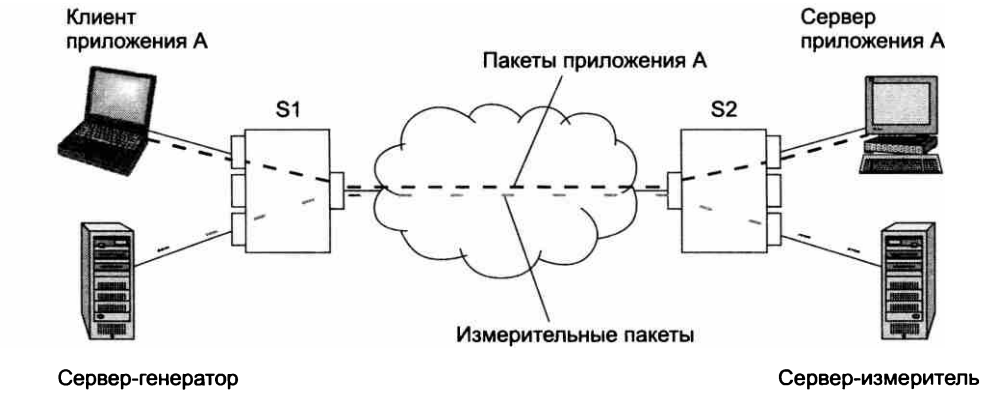


Рис. 5.4. Схема активных измерений

Возникает естественный вопрос: зачем нужно решать столько лишних проблем: размещать дополнительное оборудование, создавать условия для измерительных пакетов, близкие к условиям обработки оригинальных пакетов, и в то же время стараться не изменить нагрузку сети? Не проще ли измерять параметры реальных пакетов? Ответ заключается в том, что *активная схема упрощает процесс проведения измерений и позволяет добиться их высокой точности.*

Во-первых, так как сервер-генератор создает измерительные пакеты, то он легко может использовать специальный формат пакетов для того, чтобы поместить в них необходимую для измерения информацию, например временну́ю отметку (time-stamp) отправки пакета. Затем сервер-измеритель использует эту временну́ю отметку для вычисления времени задержки.

Во-вторых, очевидно, что для того, чтобы измерения задержки были точными, нужна хорошая синхронизация сервера-генератора и сервера-измерителя. Так как в схеме активных измерений они представляют собой выделенные узлы, такой синхронизации добиться проще, чем в случае синхронизации клиентской и серверной частей приложения *A*, которые чаще всего установлены на обычных компьютерах.

В-третьих, иногда у инженеров, проводящих измерения, просто нет доступа к компьютерам, на которых работают приложения, чтобы установить там программное обеспечение для требуемых измерений.

В-четвертых, если такой доступ и существует, то операционные системы клиента и сервера и их аппаратная платформа, скорее всего, не оптимизированы для точных измерений временны́х интервалов, а значит, вносят большие искажения в результаты (например, за счет задержек программы измерений в очереди к центральному процессору).

Однако преимущества активной схемы измерений не являются абсолютными. В некоторых ситуациях более предпочтительной является схема пассивных измерений.

Пассивные измерения основаны на измерениях характеристик реального трафика. Эту схему иллюстрирует рис. 5.5.

Приводя аргументы в пользу схемы активных измерений, мы, в сущности, описали проблемы, которые приходится решать при использовании схемы пассивных измерений: сложности синхронизации клиента и сервера, дополнительные и неопределенные

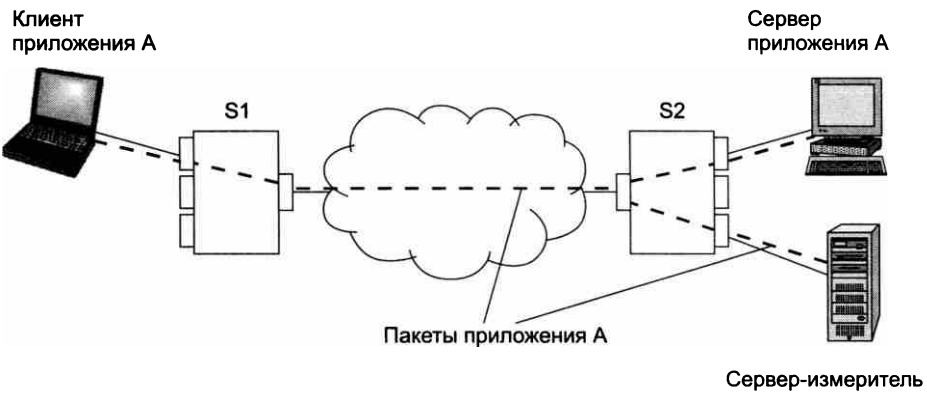


Рис. 5.5. Схема пассивных измерений

задержки, вносимые универсальными мультипрограммными операционными системами этих компьютеров, отсутствие в заголовке используемых приложением пакетов поля для переноса по сети временно́й отметки.

Частично эти проблемы решаются за счет применения отдельного сервера-измерителя. Этот сервер принимает тот же входной поток пакетов, что и один из узлов, участвующий в обмене пакетами, характеристики которых нужно измерить (на рисунке показан случай, когда сервер-измеритель ставится в параллель с сервером приложения *A*). Для того чтобы сервер-измеритель получал тот же входной поток пакетов, что и оригинальный узел, обычно прибегают к дублированию измеряемого трафика на порт, к которому подключен сервер-измеритель. Такую функцию, называемую **зеркализацией портов**, поддерживают многие коммутаторы локальных сетей. Сервер-измеритель может работать под управлением специализированной операционной системы, оптимизированной для выполнения точных измерений временны́х интервалов.

Сложнее решить проблему синхронизации. Некоторые протоколы переносят временны́е отметки в своих служебных полях, так что если, например, приложение *A* использует такой протокол, то часть проблемы решается. Однако и в этом случае остается открытым вопрос о точности системного времени в компьютере клиента приложения *A*; скорее всего, она невысока. Поэтому в пассивном режиме измеряют те характеристики, которые не требуют синхронизации передатчика и приемника, например оценивают долю потерянных пакетов.

Возможным вариантом пассивной схемы измерений является отсутствие выделенного сервера-измерителя. Некоторые приложения сами выполняют измерения задержек поступающих пакетов, например такими функциями обладают многие приложения IP-телефонии и видеоконференций, так как информация о задержках пакетов помогает определить возможную причину неудовлетворительного качества работы приложения.

ПРИМЕЧАНИЕ. СТАНДАРТЫ ИЗМЕРЕНИЙ

Как и в любой области, в сфере измерений имеются стандарты, создающие основу для одинаковой трактовки наиболее важных характеристик производительности сети. Разработкой таких стандартов занимается рабочая группа IETF под названием IPPM (IP Performance Metrics – метрики производительности IP-сетей). И хотя из названия группы видно, что ее стандарты ориентированы на характеристики именно IP-пакетов, эти стандарты носят достаточно общий характер, так что, за ис-

ключением некоторых деталей, могут применяться как основа для описания характеристик любых других протоколов (что и происходит на практике). Каждый стандарт имеет однотипную структуру. Сначала характеристика описывается как случайная величина, то есть дается определение ее единичного значения, которое является также значением ее единичного измерения. Затем дается описание того, что понимается под последовательностью замеров, то есть того, как правильно получить выборку значений характеристики. И наконец, приводятся рекомендуемые статистические оценки, которыми следует пользоваться при обработке полученной выборки значений. Обычно стандарты группы IPPM оставляют значительную свободу в выборе той или иной статистической оценки, рекомендуя несколько возможных оценок, например среднее значение, квантиль и максимальное значение.

Характеристики задержек и потерь пакетов

Для оценки производительности сети используются различные характеристики задержек и потерь пакетов, в том числе:

- односторонняя задержка пакетов;
- вариация задержки пакета;
- время реакции сети;
- время оборота пакета.

Единичное значение **односторонней задержки пакетов** (One-Way Delay Metric, OWD) определенного типа определяется как интервал времени между моментом помещения в исходящую линию связи *первого* бита пакета узлом-отправителем и моментом приема *последнего* бита пакета с входящей линии связи узла-получателя.

Под определенным типом пакета понимается пакет, который имеет некоторый заранее определенный набор признаков; ими могут быть, например, размер пакета, тип приложения, сгенерировавшего пакет, тип протокола транспортного уровня, который доставил пакет, а также некоторые другие. Цель использования набора признаков состоит в том, чтобы выделить из общего потока пакетов, приходящего в узел назначения, те пакеты, характеристики которых интересуют специалиста, проводящего измерения.

Так как в этом определении учитывается время буферизации пакета узлом-получателем, то задержка зависит от размера пакета и для получения сопоставимых результатов желательно в определении типа пакетов задавать определенный размер пакета. Определение времени задержки с учетом буферизации упрощает измерение времени прихода пакета, так как программно его можно измерить только после завершения записи всего пакета в буфер операционной системы. Кроме того, при получении только одного первого бита пакета невозможно понять, относится ли пакет к интересующему типу.

В том случае, если пакет не прибыл в узел назначения за некоторое достаточно большое время (точное значение определяет разработчик системы измерений), пакет считается утерянным, а его задержка неопределенной (ее можно полагать равной бесконечности).

Последовательность замеров рекомендуется выполнять в случайные моменты времени, подчиняющиеся распределению Пуассона. Такой порядок выбора времени замеров позволяет избежать возможной синхронизации измерений с любыми периодическими флюктуациями в поведении сети, так как подобная синхронизация может существенно исказить наблюдаемую картину.

Для одностороннего времени задержки стандарт рекомендует использовать следующие *статистические оценки*:

- квантиль для некоторого процента;
- среднее значение задержки;
- минимальное значение задержки (в выборке).

Квантили удобны для оценки задержек в тех случаях, когда процент потерь пакетов достаточно высок, так что вычисление среднего значения задержки вызывает определенные трудности (можно игнорировать потери пакетов, но тогда мы получим слишком заниженную оценку). Для вычисления квантиля потерянные пакеты можно рассматривать как пакеты, пришедшие с бесконечно большой задержкой, которая, естественно, больше значения квантиля.

Вариация задержки пакета (IP Packet Delay Variation, IPDV), которую также называют **джиттером** (jitter), очень важна для некоторых приложений. Так, при воспроизведении видеоклипа сама по себе задержка не очень существенна, например если все пакеты задерживаются ровно на десять секунд, то качество воспроизведения не пострадает, а тот факт, что картинка появляется чуть позже, чем ее отослал сервер, пользователь даже не заметит (однако в интерактивных видеоприложениях, таких как видеоконференции, подобная задержка будет, конечно, уже ощутимо раздражать). А вот если задержки постоянно изменяются в пределах от нуля до 10 секунд, то качество воспроизведения клипа заметно ухудшится, для компенсации таких переменных задержек нужна предварительная буферизация поступающих пакетов в течение времени, превышающего вариацию задержки.

> *Единичное значение* оценки вариации задержки определяется стандартом как разность односторонних задержек для пары пакетов определенного типа, полученных на интервале измерений T.

Как и для односторонней задержки, тип пакета может задаваться любыми признаками, при этом размеры обоих пакетов должны быть одинаковыми. Выбор пары пакетов на интервале измерения T должен осуществляться в соответствии с некоторым заранее принятым правилом. Например, пары могут образовываться из всех последовательных пакетов, полученных на интервале; другим примером является выбор пакетов с определенными номерами в последовательности полученных пакетов, например пакетов с номерами 1, 5, 10, 15 и т. д.

Время реакции сети представляет собой интегральную характеристику производительности сети с точки зрения пользователя. Именно эту характеристику имеет в виду пользователь, когда говорит: «Сегодня сеть работает медленно».

> Время реакции сети определяется как интервал времени между отправкой запроса пользователя к какой-либо сетевой службе и получением ответа на этот запрос.

Время реакции сети можно представить в виде нескольких слагаемых, например (рис. 5.6): времени подготовки запросов на клиентском компьютере ($t_{клиент1}$), времени передачи запросов между клиентом и сервером через сеть ($t_{сеть}$), времени обработки запросов на сервере ($t_{сервер}$), времени передачи ответов от сервера клиенту через сеть (снова $t_{сеть}$) и времени обработки получаемых от сервера ответов на клиентском компьютере ($t_{клиент2}$).

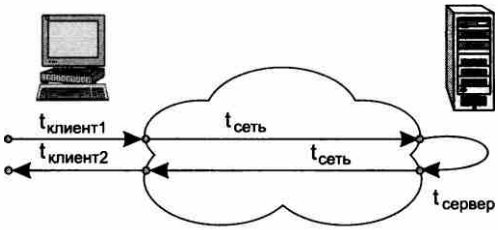


Рис. 5.6. Время реакции и время оборота

Время реакции сети характеризует сеть в целом, в том числе качество работы аппаратного и программного обеспечения серверов. Для того чтобы отдельно оценить транспортные возможности сети, используется другая характеристика — время оборота данных по сети.

Время оборота (Round Trip Time, RTT) пакета является составляющей времени реакции сети — это «чистое» время транспортировки данных от узла отправителя до узла назначения и обратно без учета времени, затраченного узлом назначения на подготовку ответа: $RTT = 2 \times t_{сеть}$.

Единичное значение времени оборота определяется как интервал времени между отправкой первого бита пакета определенного типа узлом-отправителем узлу-получателю и получением последнего бита этого пакета узлом-отправителем после того, как пакет был получен узлом-получателем и отправлен обратно.

При этом узел-получатель должен отправить пакет узлу-отправителю как можно быстрее, чтобы не вносить искажения за счет времени обработки пакета.

Последовательность замеров времени оборота рекомендуется также выполнять через случайные интервалы, подчиняющиеся распределению Пуассона.

RTT является удобной для измерений характеристикой, так как для ее получения не требуется синхронизация узла-отправителя и узла-получателя: узел-отправитель ставит временну́ю отметку на отправляемый пакет, а затем по прибытии его от узла-получателя сравнивает эту отметку со своим текущим системным временем.

Однако информативность времени оборота меньше, чем односторонней задержки, так как информация о задержке в каждом направлении теряется, а это может затруднить поиск проблемного пути в сети.

Характеристики скорости передачи

Скорость передачи данных (information rate) измеряется на каком-либо промежутке времени как частное от деления объема переданных данных за этот период на продолжительность периода. Таким образом, данная характеристика всегда по определению является средней скоростью передачи данных. Однако в зависимости от величины интервала, на котором измеряется скорость, для этой характеристики традиционно используется одно из двух наименований: средняя или пиковая скорость.

Средняя скорость передачи данных (Sustained Information Rate, SIR) — это среднесрочная характеристика. Она определяется на достаточно большом периоде времени, достаточном,

чтобы можно было говорить об устойчивом поведении такой случайной величины, которой является скорость.

Средняя скорость должна использоваться в паре с параметром, оговаривающим *период контроля* этой величины, например 10 секунд. Это означает, что скорость информационного потока вычисляется каждые 10 секунд. Если бы такие контрольные измерения не проводились, это лишило бы пользователя возможности предъявлять претензии поставщику в некоторых конфликтных ситуациях. Например, если поставщик в один из дней месяца вообще не будет передавать пользовательский трафик, а в остальные дни разрешит пользователю превышать оговоренный предел, то средняя скорость за месяц окажется в норме. В этой ситуации только регулярный контроль скорости поможет пользователю отстоять свои права.

Пиковая скорость передачи данных (Peak Information Rate, PIR) — это наибольшая скорость, которую разрешается достигать пользовательскому потоку в течение оговоренного небольшого периода времени T.

Этот период обычно называют **периодом пульсации**. Как правило, он выбирается существенно меньшим, чем период измерения средней скорости передачи. Очевидно, что при передаче трафика можно говорить об этой величине только с некоторой степенью вероятности. Например, требование к этой характеристике может быть сформулировано так: «Скорость информации не должна превышать 2 Мбит/с на периоде времени 10 мс с вероятностью 0,95». Часто значение вероятности опускают, подразумевая близость ее к единице. Пиковая скорость является краткосрочной характеристикой, она позволяет оценить способность сети справляться с пиковыми нагрузками, характерными для пульсирующего трафика и приводящими к перегрузке. Если в SLA оговорены обе скорости (SIR и PIR), очевидно, что периоды пульсации должны сопровождаться периодами относительного «затишья», когда скорость падает ниже средней. В противном случае показатель средней скорости соблюдаться не будет.

Величина пульсации (обычно обозначаемая B) служит для оценки емкости буфера коммутатора, необходимого для хранения данных во время перегрузки.

Величина пульсации равна общему объему данных, поступающих на коммутатор в течение разрешенного интервала T (периода пульсации) передачи данных с пиковой скоростью (PIR): $B = \text{PIR} \times T$.

Еще одной характеристикой скорости передачи является **коэффициент пульсации трафика** — это отношение максимальной скорости на каком-либо небольшом периоде времени к средней скорости трафика, измеренной на длительном периоде времени. Неопределенность временны́х периодов делает коэффициент пульсации *качественной* характеристикой трафика.

Скорость передачи данных можно измерять между любыми двумя узлами, или точками, сети, например между клиентским компьютером и сервером, между входным и выходным портами маршрутизатора. Для анализа и настройки сети очень полезно знать данные о пропускной способности отдельных элементов сети.

Из-за последовательного характера передачи данных различными элементами сети общая пропускная способность любого составного пути в сети будет равна *минимальной* из пропускных способностей составляющих элементов маршрута.

Для повышения пропускной способности составного пути необходимо в первую очередь обратить внимание на самые медленные элементы, называемые **узкими местами** (bottleneck).

Надежность

Характеристики потерь пакетов

В качестве характеристики потерь пакетов используется **доля потерянных пакетов** (обозначим ее L), равная отношению количества потерянных пакетов (N_L) к общему количеству переданных пакетов (N):

$$L = N_L/N.$$

Может также использоваться аналогичная характеристика, оперирующая не количествами потерянных и переданных пакетов, а объемами данных, содержавшихся в этих пакетах.

Доступность и отказоустойчивость

Для описания надежности отдельных устройств служат такие показатели надежности, как **среднее время наработки на отказ**, **вероятность отказа**, **интенсивность отказов**. Однако эти показатели пригодны только для оценки надежности простых элементов и устройств, которые при отказе любого своего компонента переходят в неработоспособное состояние. Сложные системы, состоящие из многих компонентов, могут при отказе одного из компонентов сохранять свою работоспособность. В связи с этим для оценки надежности сложных систем применяется другой набор характеристик.

Доступность (availability) означает долю времени, в течение которого система или служба находится в работоспособном состоянии.

Доступность является долговременной статистической характеристикой, поэтому измеряется на большом промежутке времени, которым может быть день, месяц или год. Примером высокого уровня доступности является коммуникационное оборудование телефонных сетей, лучшие представители которого обладают так называемой доступностью «пять девяток». Это означает, что доступность равна 0,99999, что соответствует чуть более чем пяти минутам простоя в год. Оборудование и услуги передачи данных только стремятся к такому рубежу, но рубеж трех девяток уже достигнут. Доступность услуги является универсальной характеристикой, которая важна как пользователям, так и поставщикам услуг.

Еще одной характеристикой надежности сложных систем является **отказоустойчивость** (fault tolerance). Под отказоустойчивостью понимается способность системы скрывать от пользователя отказ отдельных ее элементов.

Например, если коммутатор оснащен двумя коммутационными центрами, работающими параллельно, то отказ одного их них не приведет к полному останову коммутатора. Однако производительность коммутатора снизится, он будет обрабатывать пакеты вдвое медленнее. В отказоустойчивой системе отказ одного из ее элементов приводит к некоторому снижению качества ее работы — **деградации**, а не к полному останову. В качестве еще одного примера можно назвать использование двух физических каналов для соединения коммутаторов. В нормальном режиме работы трафик передается по двум каналам со скоростью *C* Мбит/с, а при отказе одного из них трафик будет продолжать передаваться, но уже со скоростью *C*/2 Мбит/с. Однако из-за того, что во многих случаях количественно определить степени деградации системы или услуги достаточно сложно, отказоустойчивость чаще всего применяется как качественная характеристика.

Характеристики сети поставщика услуг

Рассмотрим основные характеристики, которыми оперирует поставщик услуг, оценивая эффективность своей сети. Эти характеристики — расширяемость, масштабируемость, управляемость и совместимость — являются качественными, то есть не могут быть выражены числами и соотношениями.

Термины «расширяемость» и «масштабируемость» иногда неверно используют как синонимы.

Расширяемость означает возможность сравнительно простого добавления отдельных компонентов сети (пользователей, компьютеров, приложений, служб), наращивания длины сегментов кабелей и замены существующей аппаратуры более мощной.

При этом принципиально важно, что простота расширения системы иногда может обеспечиваться в *определенных пределах*. Например, локальная сеть Ethernet, построенная на основе одного разделяемого сегмента коаксиального кабеля, обладает хорошей расширяемостью в том смысле, что позволяет легко подключать новые станции. Однако такая сеть имеет ограничение на число станций — оно не должно превышать 30–40. Хотя сеть допускает физическое подключение к сегменту и большего числа станций (до 100), при этом резко снижается производительность сети. Наличие такого ограничения и является признаком плохой масштабируемости системы при ее хорошей расширяемости.

Масштабируемость означает, что сеть позволяет наращивать количество узлов и протяженность связей в очень широких пределах, при этом производительность сети не снижается.

Для обеспечения масштабируемости сети приходится применять дополнительное коммуникационное оборудование и специальным образом структурировать сеть. Обычно масштабируемое решение обладает многоуровневой иерархической структурой, которая позволяет добавлять элементы на каждом уровне иерархии без изменения главной идеи проекта. Примером хорошо масштабируемой сети является Интернет, технология которого (TCP/IP) оказалась способной поддерживать сеть в масштабах земного шара.

Не только сама сеть должна быть масштабируемой, но и устройства, работающие на магистрали сети, также должны обладать этим свойством, так как рост сети не должен приводить к необходимости постоянной смены оборудования. Поэтому магистральные коммутаторы и маршрутизаторы строятся обычно по модульному принципу, позволяя наращивать количество интерфейсов и производительность обработки пакетов в широких пределах.

Управляемость сети подразумевает возможность централизованно контролировать состояние основных элементов сети, выявлять и разрешать проблемы, возникающие при работе сети, анализировать производительность и планировать развитие сети.

Управляемость предполагает наличие в сети некоторых автоматизированных *средств администрирования*, которые взаимодействуют с программным и аппаратным обеспечением сети с помощью коммуникационных протоколов. В идеале средства администрирования сети обеспечивают *наблюдение* и *контроль* за каждым элементом сети и, обнаружив проблему, активизируют определенное действие, например исправляют ситуацию и уведомляют администратора о том, что произошло и какие шаги предприняты. Одновременно с этим система администрирования должна *накапливать данные*, на основании которых можно планировать развитие сети. Наконец, система администрирования должна быть независима от производителя и обладать удобным интерфейсом, позволяющим выполнять все действия с одной консоли.

Решая тактические задачи, администраторы и технический персонал сталкиваются с ежедневными проблемами поддержания работоспособности сети. Эти задачи требуют быстрого решения, обслуживающий сеть персонал должен оперативно реагировать на сообщения о неисправностях, поступающие от пользователей или автоматических средств администрирования сети. Постепенно становятся заметными более общие проблемы производительности, конфигурирования сети, обработки сбоев и безопасности данных, требующие стратегического подхода, то есть *планирования* сети. Планирование, кроме того, подразумевает умение прогнозировать изменения в требованиях пользователей к сети, решение вопросов о применении новых приложений, новых сетевых технологий и т. п.

Полезность систем администрирования особенно ярко проявляется в больших сетях: корпоративных или публичных глобальных. Без систем администрирования в таких сетях потребовалось бы содержание огромного штата обслуживающего персонала.

Однако в настоящее время большинство существующих средств вовсе не управляет сетью, а всего лишь обеспечивает *наблюдение* за ее работой и *фиксацию* важных событий, например отказов устройств. Реже системы администрирования выполняют активные действия, автоматически ликвидируя последствия нежелательного события или предотвращая его.

Совместимость, или **интегрируемость**, сети означает, что сеть способна включать в себя самое разнообразное программное и аппаратное обеспечение, то есть в ней могут сосуществовать различные операционные системы, поддерживающие разные стеки коммуникационных протоколов, а также аппаратные средства и приложения от разных производителей. Сеть, состоящая из разнотипных элементов, называется неоднородной, или *гетерогенной*, а если гетерогенная сеть работает без проблем, то она является *интегрированной*. Основной путь построения интегрированных сетей — использование модулей, выполненных в соответствии с открытыми стандартами и спецификациями.

Выводы

Главным требованием, предъявляемым к компьютерной сети, является обеспечение высокого качества предоставляемых сетью услуг. В широком понимании в понятие «качество обслуживания» включают все возможные характеристики услуг и сети, желательные для пользователя. Наиболее важные формализованные характеристики сети относятся к ее производительности и надежности.

Производительность сети оценивается с помощью статистических характеристик двух типов: характеристик скорости передачи информации и характеристик задержек передачи пакетов. В первую группу входят средняя и максимальная скорости на периоде пульсации, а также длительность этого периода. Во вторую группу входят: средняя величина задержки, средняя вариация задержки (джиттер), коэффициент вариации, а также максимальные значения задержки и вариации задержки.

Для оценки надежности сетей применяются различные характеристики, в том числе: доля потерь пакетов, коэффициент доступности, означающий долю времени, в течение которого система может быть использована, отказоустойчивость — способность системы работать в условиях отказа некоторых ее элементов.

Надежность транспортных услуг сети обеспечивается надежностью ее компонентов (каналов и коммуникационного оборудования), наличием альтернативных маршрутов, а также повторной передачей потерянных или искаженных пакетов.

Особую важность для поставщика услуг представляют такие качественные характеристики сети, как ее масштабируемость, расширяемость и управляемость.

Контрольные вопросы

1. Могут ли различаться краткосрочные и долгосрочные значения одной и той же характеристики, например средней скорости потока?
2. Какие составляющие задержки пакета являются фиксированными для пакета фиксированной длины?
3. Может ли трафик передаваться с большими задержками, но без джиттера?
4. Чем «расширяемость» сети отличается от «масштабируемости»?
5. Является ли коэффициент пульсации трафика количественной характеристикой?

Глава 6 Методы обеспечения качества обслуживания

Обзор методов обеспечения качества обслуживания

Методы обеспечения **качества обслуживания** (Quality of Service, QoS) занимают сегодня важное место в арсенале технологий сетей с коммутацией пакетов, так как они обеспечивают устойчивую работу современных мультимедийных приложений, таких как IP-телефония, видео- и радиовещание, интерактивное дистанционное обучение и т. п. В методах обеспечения качества обслуживания используются различные алгоритмы управления очередями, резервирования и обратной связи, позволяющие снизить негативные последствия временных перегрузок, возникающих в сетях с коммутацией пакетов. С их помощью проектировщики и администраторы сети могут уменьшить задержки, вариации задержек, а также потери пакетов в периоды перегрузки сети, создавая тем самым необходимые условия для удовлетворительного обслуживания сетью трафика приложений.

Очереди являются неотъемлемым атрибутом сетей с коммутацией пакетов. Сам принцип работы таких сетей подразумевает наличие буфера у каждых входного и выходного интерфейсов коммутатора пакетов. Буферизация пактов во время перегрузок представляет собой основной механизм поддержания пульсирующего трафика, обеспечивающий *высокую производительность* сетей этого типа. (Как вы знаете, в сетях с другим типом коммутации, а именно в сетях с коммутацией каналов, промежуточная буферизация данных не поддерживается.)

В то же время очереди означают неопределенную задержку при передаче пакетов через сеть, а в некоторых случаях и потери пакетов из-за переполнения буфера коммутатора или маршрутизатора, отведенного под очередь. Задержки и потери пакетов являются серьезной проблемой для некоторых видов трафика.

Таким образом, операторам пакетных сетей, весьма заинтересованным в передаче пульсирующего трафика, необходимы средства достижения *компромисса* между предельной загрузкой своей сети и требуемым клиентами качеством обслуживания их трафика.

К характеристикам качества обслуживания относят:

- одностороннюю (от отправителя к получателю) и двустороннюю (от отправителя к получателю и обратно) задержку пакетов;
- вариацию задержек пакетов;
- потери пакетов.

В методах QoS используется достаточно широкий набор механизмов, направленных на снижение негативных последствий пребывания пакетов в очередях с сохранением в то же время положительной роли очередей. Большинство из них учитывает факт существова-

ния в сети трафика различного типа, а именно то, что каждый тип трафика предъявляет разные требования к характеристикам производительности и надежности сети. Однако добиться одновременного соблюдения *всех* характеристик QoS для *всех* видов трафика весьма сложно.

Одним из наиболее значимых факторов, влияющих на характеристики качества обслуживания, является уровень загрузки сети трафиком, то есть *уровень использования пропускной способности линий связи* сети. Напомним, что пропускная способность — это характеристика физического канала, которая представляет собой максимально возможную скорость передачи информации по этому каналу (см. раздел «Характеристики физических каналов» в главе 2).

Пропускную способность сети изменить непросто, так как она определяется быстродействием интерфейсов коммуникационного оборудования и качеством линий связи, их соединяющих. Повышение пропускной способности сети — это дорогостоящая операция, связанная с заменой оборудования, которую операторы сетей проводят не очень часто, раз в несколько лет.

Если уровень использования пропускной способности постоянно является достаточно низким, то трафик всех приложений обслуживается с высоким качеством бóльшую часть времени (хотя кратковременные перегрузки сети, приводящие к задержкам и потерям пакетов, все равно возможны, но они случаются очень редко). Такое состояние сети называется «недогруженным» или же используется термин **сеть с избыточной пропускной способностью** (англоязычный термин overprovisioning). Постоянно поддерживать все части сети в недогруженном состоянии весьма дорого, но для наиболее ответственной части сети, такой как магистраль, подобный подход применяется и связан с постоянным слежением за уровнем загрузки каналов магистрали и обновлением оборудования с более высокой пропускной способностью по мере приближения загрузки к критическому уровню.

Методы QoS основаны на другом подходе, а именно на тонком *перераспределении* имеющейся пропускной способности между трафиком различного типа в соответствии с требованиями приложений. Очевидно, что эти методы усложняют сетевые устройства, которые теперь должны «знать» требования всех классов трафика, уметь их классифицировать и распределять пропускную способность сети между ними. Последнее свойство обычно достигается за счет использования для каждого выходного интерфейса коммуникационного устройства нескольких очередей пакетов вместо одной очереди; при этом в очередях применяют различные алгоритмы обслуживания пакетов, чем и достигается дифференцированное обслуживание трафика различных классов. Поэтому методы QoS часто ассоциируются с *техникой управления очередями*.

Помимо собственно техники организации очередей к методам QoS относят методы контроля параметров потока трафика, так как для гарантированно качественного обслуживания нужно быть уверенными, что обслуживаемые потоки соответствуют определенному профилю. Эта группа методов QoS получила название *методов кондиционирования трафика*.

Особое место занимают *методы обратной связи*, которые предназначены для уведомления источника трафика о перегрузке сети. Эти методы рассчитаны на то, что при получении уведомления источник снизит скорость выдачи пакетов в сеть и тем самым ликвидирует причину перегрузки.

К методам QoS тесно примыкают *методы инжиниринга трафика*. Согласно методам инжиниринга трафика маршруты передачи данных управляются таким образом, чтобы

обеспечить сбалансированную загрузку всех ресурсов сети и исключить за счет этого перегрузку коммуникационных устройств и образование длинных очередей.

Приложения и качество обслуживания

Поскольку существующие приложения в общем случае предъявляют разные требования к качеству обслуживания, важной задачей является их классификация в этом отношении. В качестве основных критериев классификации приложений используются три характеристики порождаемого ими трафика:

- относительная предсказуемость скорости передачи данных;
- чувствительность трафика к задержкам пакетов;
- чувствительность трафика к потерям и искажениям пакетов.

Предсказуемость скорости передачи данных

В отношении предсказуемости скорости передачи данных приложения делятся на два больших класса: приложения с потоковым трафиком и приложения с пульсирующим трафиком.

Приложения с потоковым трафиком (stream) порождают равномерный поток данных, который поступает в сеть с **постоянной битовой скоростью** (Constant Bit Rate, CBR). В случае коммутации пакетов трафик таких приложений представляет собой последовательность пакетов одинакового размера (равного B бит), следующих друг за другом через один и тот же интервал времени T (рис. 6.1, *а*).

CBR потокового трафика может быть вычислена путем усреднения на интервале:

$$\text{CBR} = B/T \text{ бит/с}.$$

В общем случае постоянная битовая скорость потокового трафика меньше *номинальной битовой скорости протокола*[1], который передает данные, так как между пакетами существуют паузы.

Приложения с пульсирующим трафиком (burst) отличаются высокой степенью непредсказуемости, в этих приложениях периоды молчания сменяются пульсациями, в течение которых пакеты «плотно» следуют друг за другом. В результате трафик характеризуется **переменной битовой скоростью** (Variable Bit Rate, VBR), что иллюстрирует рис. 6.1, *б*. Так, при работе приложений файлового сервиса интенсивность трафика, генерируемого приложением, может падать до нуля, когда файлы не передаются, и повышаться до максимально доступной интенсивности, ограниченной только возможностями сети, когда файловый сервер передает файл.

На рисунке показана ситуация, когда на периоде длительностью $5T$ было передано три пакета (как и на рис. 6.1, *а*, все пакеты имеют одинаковый размер B), затем на периоде длительностью T было передано 5 пакетов, а на периоде длительностью $6T$ — 2 пакета.

Пиковой скоростью трафика является скорость на втором периоде, когда за время T было передано 5 пакетов, поэтому $\text{PIR} = 5B/T$.

---

[1] Например, номинальная скорость протокола Ethernet равна 10 Мбит/с.

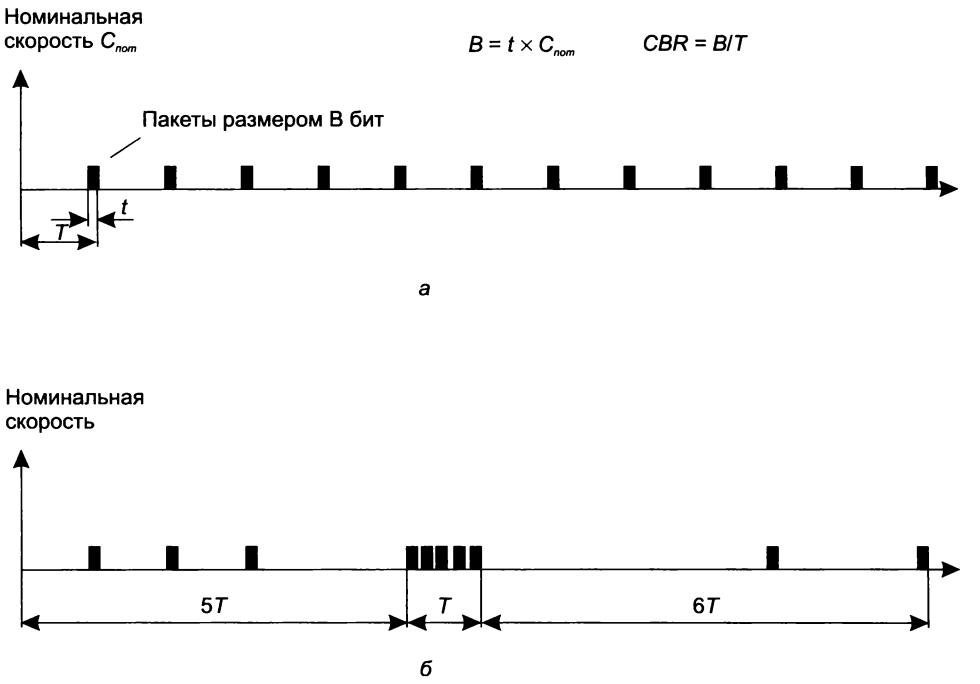


Рис. 6.1. Потоковый (*а*) и пульсирующий (*б*) трафики

В то же время средняя скорость передачи данных (Sustained Information Rate, SIR) на всех периодах наблюдений составила $10B/12T = 5B/6T$.

Для приведенного примера можно подсчитать коэффициент пульсации. Напомним, что согласно определению, данному в главе 5, он равен отношению пиковой скорости на каком-либо небольшом периоде времени к средней скорости трафика, измеренной на длительном периоде времени. Таким образом, коэффициент пульсации равен PIR/SIR = $=(5B/T)/(5B/6T) = 6$.

Практически любой трафик, даже трафик потоковых приложений, имеет ненулевой коэффициент пульсации. Просто значения коэффициентов пульсации у потокового и пульсирующего трафиков существенно различаются. У приложений с пульсирующим трафиком он обычно находится в пределах от 2 до 100, а у потоковых приложений он близок к 1. В локальных сетях коэффициент пульсации обычно выше, чем в глобальных, поскольку на магистралях глобальных сетей трафик представляет собой сумму трафиков многих источников, что по закону больших чисел приводит к сглаживанию результирующего трафика.

Чувствительность трафика к задержкам пакетов

Еще один критерий классификации приложений по типу трафика — их чувствительность к задержкам пакетов и их вариациям. Далее перечислены основные типы приложений в порядке повышения чувствительности к задержкам пакетов.

- **Асинхронные приложения** практически не имеют ограничений на время задержки (эластичный трафик). Пример такого приложения — электронная почта.
- **Интерактивные приложения.** Задержки могут быть замечены пользователями, но они не сказываются негативно на функциональности приложений. Пример — текстовый редактор, работающий с удаленным файлом.
- **Изохронные приложения** имеют порог чувствительности к вариациям задержек, при превышении которого резко снижается функциональность приложений. Пример — передача голоса, когда при превышении порога вариации задержек в 100–150 мс резко снижается качество воспроизводимого голоса.
- **Сверхчувствительные к задержкам приложения.** Задержка доставки данных сводит функциональность приложения к нулю. Пример — приложения, управляющие техническим объектом в реальном времени. При запаздывании управляющего сигнала на объекте может произойти авария.

Вообще говоря, интерактивность приложения всегда повышает его чувствительность к задержкам. Например, широковещательная рассылка аудиоинформации может выдерживать значительные задержки в передаче пакетов (оставаясь чувствительным к вариациям задержек), а интерактивный телефонный или телевизионный разговор их не терпит, что хорошо заметно при трансляции разговора через спутник. Длительные паузы в разговоре вводят собеседников в заблуждение, часто они теряют терпение и начинают очередную фразу одновременно.

Наряду с приведенной классификацией, тонко дифференцирующей чувствительность приложений к задержкам и их вариациям, существует и более грубое деление приложений по этому признаку на два класса: асинхронные и синхронные. К *асинхронным* относят те приложения, которые нечувствительны к задержкам передачи данных в очень широком диапазоне, вплоть до нескольких секунд, а все остальные приложения, на функциональность которых задержки влияют существенно, относят к *синхронным* приложениям.

Интерактивные приложения могут относиться как к асинхронным (например, текстовый редактор), так и к синхронным (например, видеоконференция).

Чувствительность трафика к потерям и искажениям пакетов

И наконец, последним критерием классификации приложений является их чувствительность к потерям пакетов. Здесь обычно делят приложения на две группы.

- **Приложения, чувствительные к потере данных.** Практически все приложения, передающие алфавитно-цифровые данные (к которым относятся текстовые документы, коды программ, числовые массивы и т. п.), обладают высокой чувствительностью к потере отдельных, даже небольших фрагментов данных. Такие потери часто ведут к полному обесцениванию остальной успешно принятой информации. Например, отсутствие хотя бы одного байта в коде программы делает ее совершенно неработоспособной. Все традиционные сетевые приложения (файловый сервис, сервис баз данных, электронная почта и т. д.) относятся к этому типу приложений.
- **Приложения, устойчивые к потере данных.** К этому типу относятся многие приложения, передающие трафик с информацией об инерционных физических процессах.

Устойчивость к потерям объясняется тем, что небольшое количество отсутствующих данных можно определить на основе принятых. Так, при потере одного пакета, несущего несколько последовательных замеров голоса, отсутствующие замеры при воспроизведении голоса могут быть заменены аппроксимацией на основе соседних значений. К такому типу относится большая часть приложений, работающих с мультимедийным трафиком (аудио- и видеоприложения). Однако устойчивость к потерям имеет свои пределы, поэтому процент потерянных пакетов не может быть большим (например, не более 1 %). Можно отметить также, что не любой мультимедийный трафик устойчив к потерям данных, так, компрессированный голос и видеоизображение очень чувствительны к потерям, поэтому относятся к первому типу приложений.

Управление очередями

Определить основные характеристики QoS и сформулировать требования к ним — значит наполовину решить задачу. Пользователь формулирует свои требования к качеству обслуживания в виде некоторых предельных значений характеристик QoS, которые не должны быть превышены, например он может указать, что предельное значение вариации задержки пакетов не должно превышать 50 мс с вероятностью 0,99. Но как заставить сеть справиться с поставленной задачей? Какие меры нужно предпринять, чтобы вариации задержек действительно не превысили эту величину?

Для понимания механизмов поддержки QoS полезно исследовать процесс образования очередей на сетевых устройствах и понять наиболее существенные факторы, влияющие на длину очереди.

Анализ очередей

Существует ветвь прикладной математики, предметом которой являются процессы образования очередей. Эта дисциплина так и называется — **теория очередей**. Мы не будем углубляться в математические основы этой теории, приведем только некоторые ее выводы, существенные для рассматриваемой нами проблемы QoS.

Теория очередей рассматривает временны́е процессы образования очередей в буфере абстрактного устройства, в который поступает случайный поток абстрактных заявок на обслуживание. Модели теории очередей позволяют оценить среднюю длину очереди в буфере и среднее время ожидания заявки в очереди в зависимости от характеристик входного потока и времени обслуживания.

При применении теории очередей к анализу процессов, происходящих в компьютерных сетях, заявками на обслуживание являются пакеты данных, а обслуживающими устройствами — интерфейсы сетевых устройств, таких как коммутаторы и маршрутизаторы (рис. 6.2).

Среднее время обслуживания заявки μ соответствует среднему времени продвижения пакета процессором коммутатора из входного буфера в выходной канал.

Модели теории очередей весьма упрощенно описывают процессы, происходящие в коммутаторе или маршрутизаторе. Тем не менее они полезны для понимания основных факторов, влияющих на величину очереди в буфере сетевого устройства и среднее время пребывания пакета в буфере.

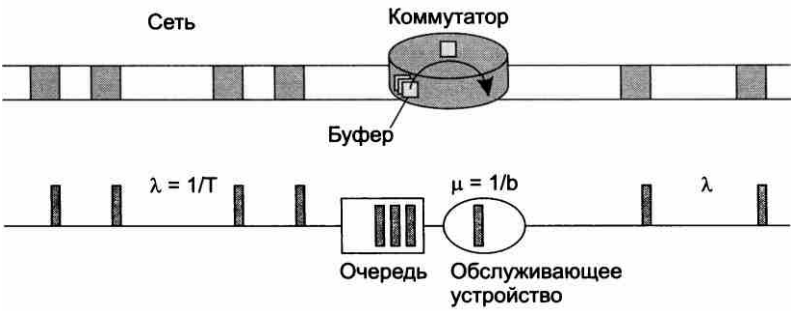


Рис. 6.2. Выходной интерфейс коммутатора как разделяемый ресурс

Одним из таких факторов является коэффициент загрузки (использования) интерфейса, равный отношению средней интенсивности потока поступления пакетов λ в интерфейс к среднему времени обработки пакета μ (это время включает все стадии продвижения пакета в выходной канал). В теории очередей этот коэффициент принято обозначать как ρ.

На рис. 6.3 показана зависимость среднего времени ожидания пакета в буфере w от ρ. Как видно из поведения кривой, коэффициент ρ играет ключевую роль в образовании очереди. Если значение ρ близко к нулю, то среднее время ожидания тоже очень близко к нулю. А это означает, что пакеты почти никогда не ожидают обслуживания в буфере (в момент их прихода он оказывается пустым), а сразу передаются на выход. И наоборот, если ρ приближается к 1, то время ожидания растет очень быстро и нелинейно (и в пределе равно бесконечности). Такое поведение очереди интуитивно понятно, ведь чем ближе средние значения интервалов между пакетами к среднему времени их обслуживания, тем сложнее обслуживающему устройству (интерфейсу) справляться с нагрузкой.

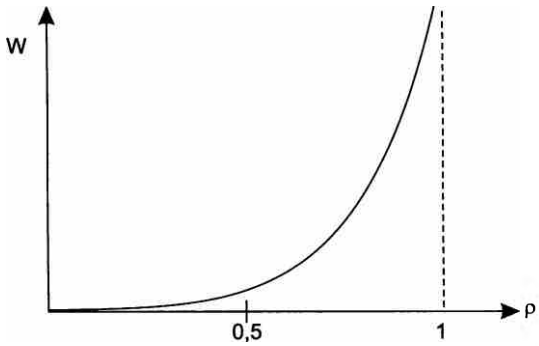


Рис. 6.3. Зависимость среднего времени ожидания заявки от коэффициента использования ресурса

Сетевые инженеры хорошо знакомы с видом графика, представленного на рис. 6.3, так как он соответствует поведению тех кривых, которые инженеры видят при анализе результатов мониторинга задержек и потерь пакетов в реальных сетях, а именно резкому ухудшению качества обслуживания при достижении коэффициента использования пропускной способности интерфейсов сети некоторого порогового значения.

В приведенном графике есть и нечто неожиданное. Трудно представить, что обслуживающее устройство (сетевой ресурс) практически перестает справляться со своими обязанностями, когда его коэффициент использования приближается к 1. Ведь в этом случае нагрузка не превышает его возможностей, а только приближается к этому пределу. Интуитивно не очень понятна также причина существования очередей при значениях ρ в окрестностях 0,5. Интенсивность обработки трафика вдвое превышает интенсивность нагрузки, а очереди существуют!

Такие парадоксальные на первый взгляд результаты характерны для систем, в которых протекают случайные процессы. Так как во внимание принимаются *средние* значения интенсивностей потоков на больших промежутках времени, то на небольших промежутках времени они могут *существенно отклоняться* от этих значений. Очередь создается на тех промежутках, на которых интенсивность поступления пакетов намного превосходит интенсивность обслуживания.

Перегрузка ресурсов может привести к полной деградации сети, когда, несмотря на то что сеть передает пакеты, полезная скорость передачи данных оказывается равной нулю. Эта ситуация имеет место, если задержки доставки всех пакетов превосходят некоторый порог и пакеты по тайм-ауту отбрасываются узлом назначения как устаревшие. Если же протоколы, работающие в сети, используют надежные процедуры передачи данных на основе квитирования и повторной передачи утерянных пакетов, то процесс перегрузки будет нарастать лавинообразно.

Существует еще один важный параметр, оказывающий непосредственное влияние на образование очередей в сетях с коммутацией пакетов. Этим параметром является вариация интервалов входного потока пакетов, то есть пульсация входного трафика.

На рис. 6.4 показано семейство зависимостей w от ρ, полученных для разных значений коэффициента вариации CV входного потока пакетов. Из рисунка видно, что чем меньше пульсирует входной поток (CV приближается к нулю), тем меньше проявляется эффект лавинообразного образования очереди при приближении коэффициента загрузки ресурса к 1. И наоборот, чем больше CV, тем раньше (при меньших значениях ρ) начинает проявляться этот эффект.

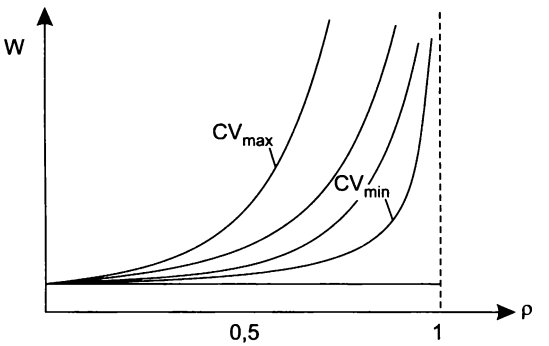


Рис. 6.4. Влияние степени пульсации потока на задержки

Из поведения графиков на рисунке можно сделать два вывода: во-первых, для оценки значений задержек в очередях на коммутаторах сети недостаточно информации о коэф-

фициенте загрузки ρ, необходимо также знать параметры пульсации трафика. Во-вторых, для снижения уровня задержек нужно снижать значение ρ и уменьшать пульсацию трафика.

Следует подчеркнуть, что модели теории очередей из-за упрощенного представления процессов, протекающих в коммуникационных устройствах сети, дают только *качественную* картину зависимостей задержек и потерь пакетов от параметров трафика и производительности устройств. Более адекватные результаты могут быть получены путем имитационного моделирования, но надежнее всего полагаться на измерение задержек и потерь пакетов в реальной сети с помощью соответствующих тестеров и систем.

Очереди и различные классы трафика

Посмотрим, как можно применить знания о зависимости поведения очередей от коэффициента загрузки для реализации основной идеи методов QoS, а именно — дифференцированного обслуживания классов трафика с различными требованиями к характеристикам производительности и надежности сети. Чтобы проще было в этом разобраться, будем пока делить все потоки на два класса — чувствительный к задержкам (трафик реального времени, например голосовой) и эластичный, допускающий большие задержки, но чувствительный к потерям данных.

Мы знаем, что если обеспечить для чувствительного к задержкам трафика коэффициент загрузки каждого ресурса не более 0,2, то, очевидно, задержки в каждой очереди будут небольшими и, скорее всего, приемлемыми для многих типов приложений этого класса. Для эластичного трафика, слабо чувствительного к задержкам, можно допустить более высокий коэффициент загрузки, но не более 0,9. Для того чтобы пакеты этого класса не терялись, нужно предусмотреть для них буферную память, достаточную для хранения всех пакетов периода пульсации. Эффект от такого распределения загрузки ресурса иллюстрирует рис. 6.5.

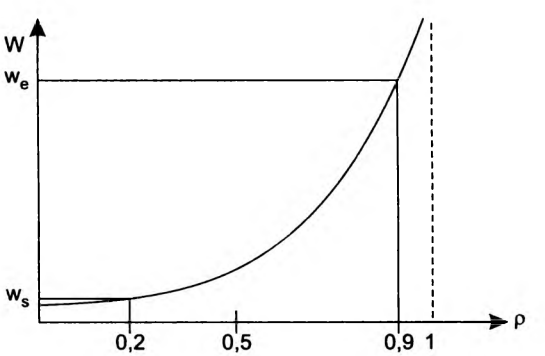


Рис. 6.5. Обслуживание эластичного и чувствительного к задержкам трафика

Задержки чувствительного к задержкам трафика равны w_s, а задержки эластичного трафика — w_e.

Чтобы добиться различных коэффициентов использования ресурсов для разных классов трафика, нужно в каждом коммутаторе для каждого ресурса поддерживать две разные

очереди. Алгоритм выборки пакетов из очередей должен отдавать предпочтение очереди чувствительных к задержкам пакетов. Если бы все пакеты первой очереди обслуживались приоритетно, а пакеты второй очереди — только тогда, когда первая очередь пуста, то для трафика первой очереди трафик второй очереди фактически перестал бы существовать. Поэтому если отношение средней интенсивности приоритетного трафика λ_1 к производительности ресурса μ равно 0,2, то и коэффициент загрузки для него равен 0,2. А вот для эластичного трафика, пакеты которого всегда ждут обслуживания приоритетных пакетов, коэффициент загрузки подсчитывается по-другому. Если средняя интенсивность эластичного трафика равна λ_2, то для него ресурс будет загружен на $(\lambda_1 + \lambda_2)/\mu$. Так что если мы хотим, чтобы для эластичного трафика коэффициент загрузки составлял 0,9, то его интенсивность должна вычисляться из соотношения $\lambda_2/\mu = 0{,}7$.

Основная идея, лежащая в основе всех методов поддержания характеристик QoS, заключается в следующем: *общая производительность каждого ресурса должна делиться между разными классами трафика неравномерно.*

Можно ввести более чем два класса обслуживания и стараться, чтобы каждый класс работал на своей части кривой задержек. Если такая задача решена, то можно обеспечить улучшение характеристик QoS за счет других методов, например снижая пульсацию трафика. Осталось выяснить, каким образом обеспечить такие условия для разных классов трафика в каждом узле сети.

Техника управления очередями

Техника управления очередями нужна для работы в периоды временных перегрузок, когда сетевое устройство не справляется с передачей пакетов на выходной интерфейс в том темпе, в котором они поступают. Если причиной перегрузки является недостаточная производительность процессорного блока сетевого устройства, то необработанные пакеты временно накапливаются во входной очереди соответствующего входного интерфейса. Очередей к входному интерфейсу может быть несколько, если мы дифференцируем запросы на обслуживание по нескольким классам. В том же случае, когда причина перегрузки заключается в ограниченной пропускной способности выходного интерфейса, пакеты временно сохраняются в выходной очереди (или очередях) этого интерфейса.

Очередь FIFO

В **очереди FIFO** в случае перегрузки все пакеты помещаются в *одну* общую очередь и выбираются из нее в том порядке, в котором поступили. Во всех устройствах с коммутацией пакетов алгоритм FIFO используется по умолчанию, так что такая очередь также обычно называется очередью «по умолчанию». Достоинствами этого подхода являются простота реализации и отсутствие потребности в конфигурировании. Однако ему присущ и коренной недостаток — *невозможность дифференцированной обработки пакетов различных потоков*. Все пакеты стоят в общей очереди на равных основаниях. Вместе оказываются как пакеты чувствительного к задержкам голосового трафика, так и нечувствительного к задержкам, но очень интенсивного трафика резервного копирования, длительные пульсации которого могут надолго задержать голосовой пакет.

Приоритетное обслуживание

Очереди с приоритетным обслуживанием очень популярны во многих областях вычислительной техники, в частности в операционных системах, когда одним приложениям нужно отдать предпочтение перед другими при обработке их в мультипрограммной смеси. Применяются эти очереди и для преимущественной по сравнению с другими обработки одного класса трафика.

Механизм приоритетного обслуживания основан на разделении всего сетевого трафика на небольшое количество классов и последующем назначении каждому классу некоторого числового признака — **приоритета**.

Классификация трафика представляет собой отдельную задачу. Пакеты могут разбиваться на приоритетные классы на основании различных признаков: адреса назначения, адреса источника, идентификатора приложения, генерирующего этот трафик, любых других комбинаций признаков, которые содержатся в заголовках пакетов. Правила классификации пакетов представляют собой часть политики администрирования сети.

Точка классификации трафика может размещаться в каждом коммуникационном устройстве. Более масштабируемое решение — размещение механизмов классификации трафика в одном или нескольких устройствах, расположенных на границе сети (например, в коммутаторах корпоративной сети, к которым подключаются компьютеры пользователей, или во входных маршрутизаторах сети поставщика услуг). В этом случае необходимо специальное поле в пакете, в котором можно запомнить назначенное значение приоритета, чтобы им могли воспользоваться остальные сетевые устройства, обрабатывающие трафик после классифицирующего устройства. Такое поле имеется в заголовке многих протоколов. В тех же случаях, когда специального поля приоритета в заголовке нет, разрабатывается дополнительный протокол, который вводит новый заголовок с таким полем (так произошло, например, с протоколом Ethernet — см. раздел «Виртуальные локальные сети» в главе 13).

Приоритеты могут назначаться не только коммутатором или маршрутизатором, но и приложением на узле-отправителе. Необходимо также учитывать, что если в сети отсутствует централизованная политика назначения приоритетов, каждое сетевое устройство может не согласиться с приоритетом, назначенным данному пакету в другой точке сети. В этом случае оно перепишет значение приоритета в соответствии с локальной политикой, принятой непосредственно на данном устройстве.

В сетевом устройстве, поддерживающем приоритетное обслуживание, имеется *несколько* очередей (буферов) — по одной для каждого приоритетного класса. Пакет, поступивший в период перегрузок, помещается в очередь, соответствующую его приоритетному классу[1]. На рис. 6.6 приведен пример использования четырех приоритетных очередей с высоким, средним, нормальным и низким приоритетами. До тех пор пока из более приоритетной очереди не будут выбраны все имеющиеся в ней пакеты, устройство не переходит к обработке следующей, менее приоритетной очереди. Поэтому пакеты с низким приоритетом обрабатываются только тогда, когда пустеют все вышестоящие очереди: с высоким, средним и нормальным приоритетами.

---

[1] Иногда несколько очередей изображают в виде одной очереди, в которой находятся заявки различных классов. Если заявки выбираются из очереди в соответствии с их приоритетами, то это просто другое представление одного и того же механизма.

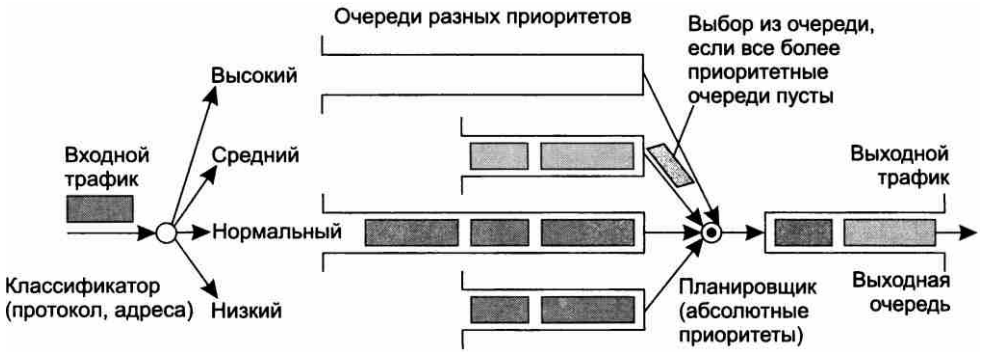


Рис. 6.6. Приоритетные очереди

Размер буфера сетевого устройства определяет максимальную длину очереди ожидающих обслуживания пакетов, если пакет поступает при заполненном буфере, то он просто отбрасывается. Обычно по умолчанию всем приоритетным очередям отводятся одинаковые буферы, но многие устройства разрешают администратору назначать каждой очереди буфер индивидуального размера. Размер буфера определяется в идеальном случае таким образом, чтобы его хватало с некоторым запасом для хранения очереди среднестатистической длины. Однако установить это значение достаточно сложно, так как оно изменяется в зависимости от нагрузки сети, поэтому требуется постоянное и длительное наблюдение за работой сети. В общем случае, чем выше значимость трафика для предприятия, чем больше его интенсивность и пульсации, тем больший размер буфера требуется этому трафику. В примере, приведенном на рис. 6.6, для трафика высшего и нормального приоритетов выбраны большие размеры буферов, а для остальных двух классов — меньшие. Мотивы принятого решения для высшего приоритета очевидны, а трафик нормального приоритета имеет высокую интенсивность и значительный коэффициент пульсаций.

Приоритетное обслуживание очередей обеспечивает высокое качество обслуживания для пакетов из самой приоритетной очереди. Если средняя интенсивность их поступления в устройство не превосходит пропускной способности выходного интерфейса (и производительности внутренних продвигающих блоков самого устройства), то пакеты высшего приоритета всегда получают ту пропускную способность, которая им нужна. Уровень задержек высокоприоритетных пакетов также минимален. Однако он не нулевой и зависит в основном от характеристик потока этих пакетов — чем выше пульсации потока и его интенсивность, тем вероятнее возникновение очереди, образованной пакетами данного высокоприоритетного потока. Трафик всех остальных приоритетных классов почти прозрачен для пакетов высшего приоритета. Слово «почти» относится к ситуации, когда высокоприоритетный пакет вынужден ждать завершения обслуживания низкоприоритетного пакета, если его приход совпадает по времени с началом продвижения низкоприоритетного пакета на выходной интерфейс. Этот эффект иллюстрирует рис. 6.7, на котором показано, что после разделения всего трафика на приоритетный и обычный (здесь имеются две очереди) коэффициент использования приоритетного трафика снизился с 50 до 15 %, так как нагрузка обычного трафика перестала влиять на использование выходного интерфейса приоритетным трафиком.

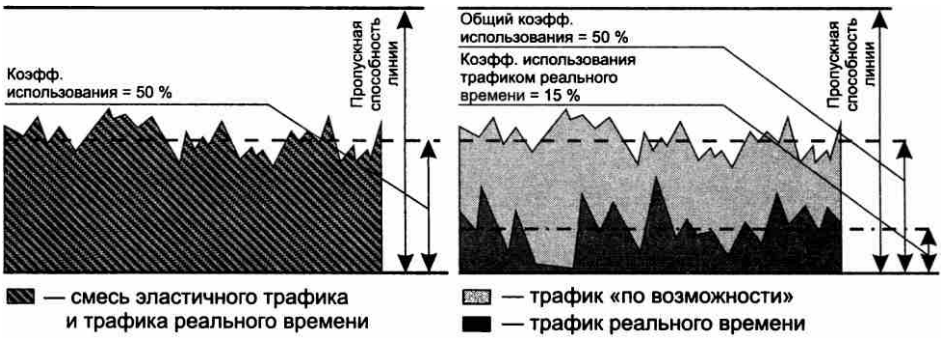


Рис. 6.7. Снижение коэффициента использования линии для приоритетного трафика: весь трафик обслуживается одной очередью (*а*); трафик реального времени обслуживается приоритетной очередью, а остальной трафик — очередью, применяемой по умолчанию (*б*)

Что же касается остальных приоритетных классов, то качество их обслуживания будет ниже, чем у пакетов самого высокого приоритета, причем уровень снижения может быть очень существенным. Если коэффициент нагрузки выходного интерфейса, определяемый только трафиком высшего приоритетного класса, приближается в какой-то период времени к единице, то трафик остальных классов на это время просто замораживается. Поэтому приоритетное обслуживание обычно применяется для чувствительного к задержкам класса трафика, имеющего небольшую интенсивность. При таких условиях обслуживание этого класса не слишком ущемляет обслуживание остального трафика. Например, голосовой трафик чувствителен к задержкам, но его интенсивность обычно не превышает 8–16 Кбит/с, так что при назначении ему высшего приоритета ущерб остальным классам трафика оказывается не очень значительным.

Взвешенные очереди

Механизм взвешенных очередей разработан для того, чтобы можно было предоставить всем классам трафика определенный минимум пропускной способности. Под *весом* данного класса понимается процент предоставляемой классу трафика пропускной способности от полной пропускной способности выходного интерфейса.

При взвешенном обслуживании, так же как и при приоритетном, трафик делится на несколько классов и для каждого класса ведется отдельная очередь пакетов. Но с каждой очередью связывается *не приоритет, а процент пропускной способности* ресурса, гарантируемый данному классу трафика при перегрузках этого ресурса. Для входного потока таким ресурсом является процессор, а для выходного (после выполнения коммутации) — выходной интерфейс.

Поясним данный алгоритм управления очередями на примере. Показанное на рис. 6.8 устройство поддерживает для пяти классов трафика пять очередей к выходному интерфейсу коммутатора. Этим очередям при перегрузках выделяется соответственно 10, 10, 30, 20 и 30 % пропускной способности выходного интерфейса.

Достигается поставленная цель за счет того, что очереди обслуживаются последовательно и циклически и в каждом цикле обслуживания из каждой очереди выбирается такое число

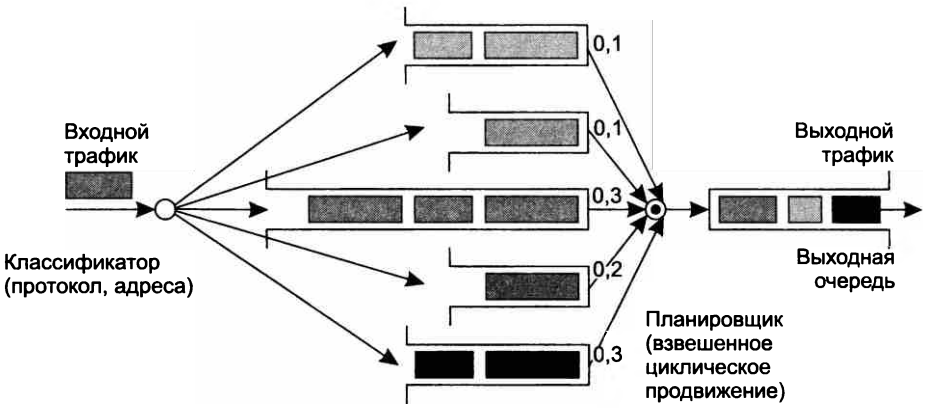


Рис. 6.8. Взвешенные очереди

байтов, которое соответствует весу данной очереди. Так, если цикл просмотра очередей в рассматриваемом примере равен одной секунде, а скорость выходного интерфейса составляет 100 Мбит/с, то при перегрузках в каждом цикле первой очереди уделяется 10 % времени, то есть 100 мс, и выбирается 10 Мбит данных, из второй — тоже 10 Мбит, из третьей — 30 Мбит, из четвертой — 20 Мбит, из пятой — 30 Мбит.

В результате каждому классу трафика достается гарантированный минимум пропускной способности, что во многих случаях является более желательным результатом, чем подавление низкоприоритетных классов высокоприоритетными.

Так как данные выбираются из очереди пакетами, а не битами, то реальное распределение пропускной способности между классами трафика всегда немного отличается от планируемого. Так, в предыдущем примере вместо 10 % первый класс трафика мог бы получать при перегрузках 9 или 12 %. Чем больше время цикла, тем точнее соблюдаются требуемые пропорции между классами трафика, так как из каждой очереди выбирается большее число пакетов и влияние размера каждого пакета усредняется.

В то же время длительный цикл приводит к большим задержкам передачи пакетов. Так, при выбранном нами для примера цикле в одну секунду задержка может составить одну и более секунд — ведь арбитр возвращается к каждой очереди не чаще чем раз в секунду, кроме того, в очереди может находиться более одного пакета. Поэтому при выборе времени цикла нужно обеспечить баланс между точностью соблюдения пропорций пропускной способности и стремлением к снижению задержки.

Для нашего примера более сбалансированным выглядит время цикла в 1000 мкс. С одной стороны, такое время гарантирует более низкий уровень задержек, так как очереди просматриваются намного чаще, чем при секундном цикле. С другой стороны, этого времени достаточно, чтобы выбрать из каждой очереди в среднем по несколько пакетов (первой очереди в нашем примере будет отводиться 100 мкс, что достаточно, например, для передачи в выходной канал одного пакета Fast Ethernet или десяти пакетов Gigabit Ethernet).

На уровень задержек и вариации задержек пакетов для некоторого класса трафика при взвешенном обслуживании в значительной степени влияет **относительный коэффициент использования**. В этом случае коэффициент подсчитывается как отношение интенсивности входного трафика класса к пропускной способности, выделенной этому классу в со-

ответствии с его весом. Например, если мы выделили первой очереди 10 % от общей пропускной способности выходного интерфейса, то есть 10 Мбит/с, а средняя интенсивность потока, который попадает в эту очередь, равна 3 Мбит/с, то коэффициент использования для этого потока составит 3/10 = 0,3. Качественное поведение очереди и соответственно задержек здесь выглядит примерно так же, как и в случае очереди FIFO — чем меньше коэффициент загрузки, тем меньше средняя длина очереди и тем меньше задержки.

Еще одним вариантом взвешенного обслуживания является **взвешенное справедливое обслуживание** (Weighted Fair Queuing, WFQ). В случае подобного обслуживания пропускная способность ресурса делится между всеми потоками поровну, то есть «справедливо».

Взвешенное обслуживание обеспечивает требуемые соотношения между интенсивностями трафика различных очередей только *в периоды перегрузок*, когда каждая очередь постоянно заполнена. Если же какая-нибудь из очередей пуста (то есть для трафика данного класса текущий период не является периодом перегрузки), то при просмотре очередей она игнорируется, а ее время обслуживания распределяется между остальными очередями в соответствии с их весом. Поэтому в отдельные периоды трафик определенного класса может обладать большей интенсивностью, чем соответствующий процент от пропускной способности выходного интерфейса.

Комбинированные алгоритмы обслуживания очередей

Каждый из описанных подходов имеет свои достоинства и недостатки. Приоритетное обслуживание, обеспечивая минимальный уровень задержек для очереди наивысшего приоритета, не дает никаких гарантий в отношении средней пропускной способности для трафика очередей более низких приоритетов. Взвешенное обслуживание обеспечивает заданное распределение средней пропускной способности, но не учитывает требований к задержкам.

Существуют **комбинированные алгоритмы обслуживания очередей**. В наиболее популярном алгоритме подобного рода поддерживается одна приоритетная очередь, а остальные очереди обслуживаются в соответствии со взвешенным алгоритмом. Обычно приоритетная очередь используется для чувствительного к задержкам трафика, а остальные — для эластичного трафика нескольких классов. Каждый класс эластичного трафика получает некоторый минимум пропускной способности при перегрузках. Этот минимум вычисляется как процент от пропускной способности, оставшейся от приоритетного трафика. Очевидно, что нужно как-то ограничить приоритетный трафик, чтобы он не поглощал всю пропускную способность ресурса. Обычно для этого применяются механизмы кондиционирования трафика, которые рассматриваются далее.

Механизмы кондиционирования трафика

Как мы помним, основной идеей методов QoS является выделение определенной доли пропускной способности определенным потокам трафика, при этом величина полученной потоком доли должна быть достаточной для того, чтобы качество обслуживания потока было удовлетворительным. Очереди с различными алгоритмами обслуживания позволяют реализовать только одну часть этой идеи — они выделяют определенную долю пропускной способности некоторому потоку пакетов. Вторая же часть задачи — обеспечение требуемого

качества обслуживания потока — решается ограничением его скорости. Скорость, а также связанный с ней относительный коэффициент использования пропускной способности не должны превышать значений, предельных для поддержания требуемого качества обслуживания. Эту задачу решают **механизмы кондиционирования трафика**, включающие классификацию, профилирование и формирование трафика.

Мы уже имели дело с **классификацией трафика**, когда при изучении приоритетных и взвешенных очередей предполагали наличие некоего механизма, решающего, какие пакеты нужно отправить в ту или иную очередь. Такого рода классификация обычно выполняется средствами фильтрации трафика, имеющимися в коммутаторах и маршрутизаторах пакетных сетей. Для классификации используются различные признаки пакетов, например адреса назначения и источника, тип протокола транспортного или прикладного уровня.

Профилирование и формирование трафика

Профилирование (policing) представляет собой меру принудительного воздействия на трафик, направленного на ограничение скорости потока пакетов. Профилирование обеспечивает соответствие потока пакетов заданному скоростному **профилю** — набору заданных параметров потока. В качестве основного параметра обычно выступает средняя скорость потока пакетов, измеренная на определенном интервале времени[1]. Пакеты, которые не укладываются в заданный профиль, либо *отбрасываются*, либо *деквалифицируются*, то есть помещаются в класс обслуживания с более низкими привилегиями, например переводятся из приоритетного класса в стандартный класс, обслуживаемый «по возможности».

Профилирование чаще всего применяют для ограничения трафика, поступающего в приоритетную очередь, так как этот механизм является единственно возможным средством предотвращения вытеснения всего остального трафика приоритетным трафиком.

Рисунок 6.9 иллюстрирует действие механизма профилирования, показывая значения скорости трафика, измеренные на достаточно малых интервалах времени до и после профилирования. Как видно из рисунка, отбрасывание пакетов при профилировании приводит к удержанию скорости потока на заданном уровне в те интервалы времени, когда скорость входящего потока превосходит этот предел, и к сохранению исходной скорости в остальные периоды.

Формирование трафика (shaping) в каком-то смысле подобно профилированию, так как имеет схожую цель — ограничение скорости трафика, или более точно — приведение параметров потока к заданному профилю. Однако достигается эта цель другим способом. Вместо того чтобы отбрасывать избыточные пакеты, то есть те, передача которых могла бы привести к превышению лимита скорости, механизм формирования трафика *задерживает* пакеты-нарушители так, что результирующая скорость оказывается в заданных пределах. Эффект формирования трафика иллюстрирует рис. 6.10. График скорости трафика сглаживается за счет «срезания» выступов (задержки пакетов, выходящих за уровень предельной скорости) и заполнения впадин (перемещения их в другие интервалы времени, в которых скорость оказывается меньше установленного предела).

---

[1] Применяются и более сложные варианты профилирования, например учитывающие среднюю и пиковые скорости.

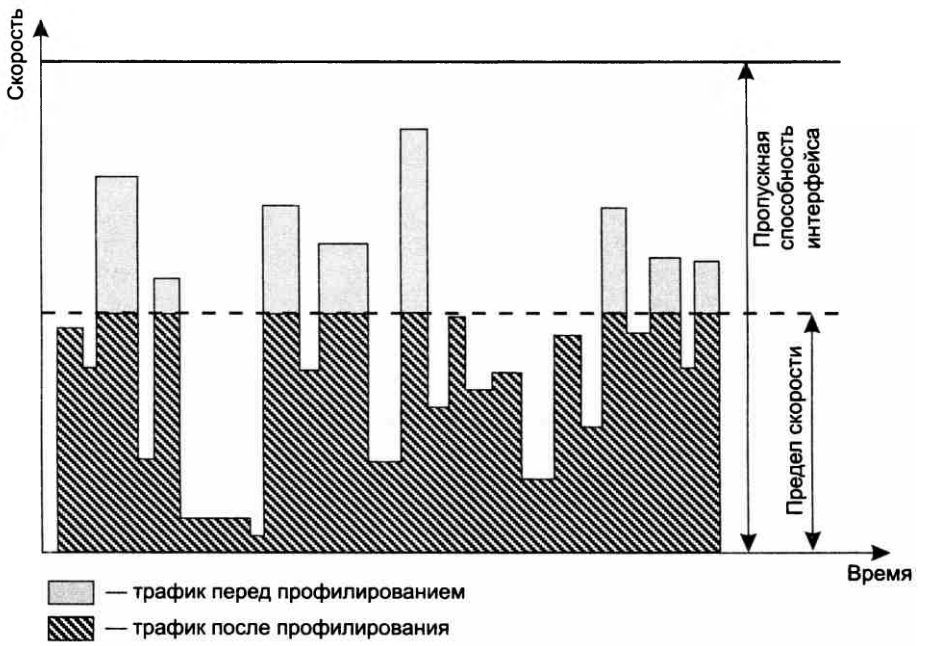


Рис. 6.9. Эффект профилирования — отбрасывание избыточного трафика

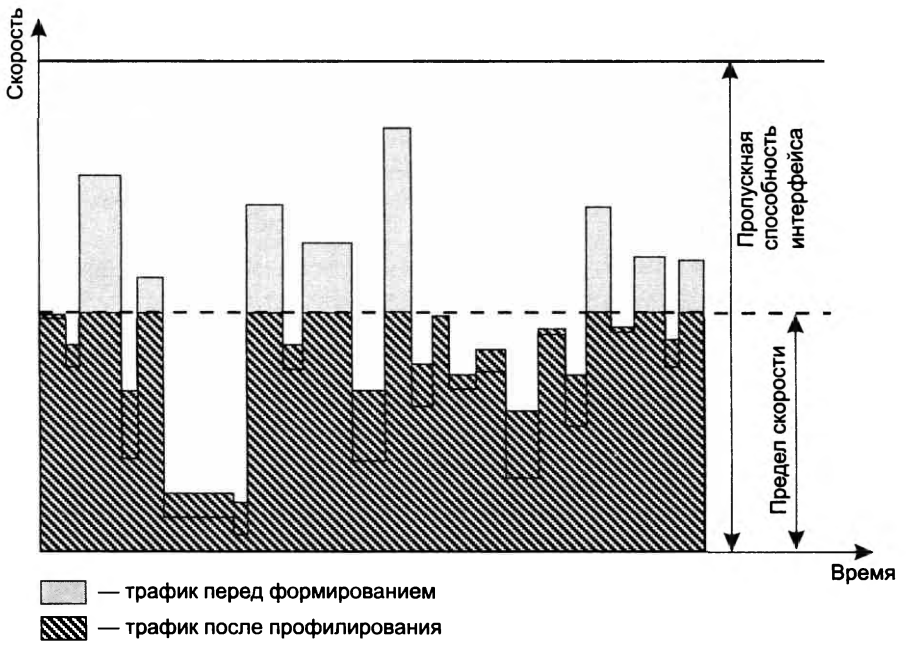


Рис. 6.10. Эффект формирования трафика — сглаживание

Обычно формирование трафика применяется к потокам, исходящим от коммутатора или маршрутизатора. Это делается в тех случаях, когда известно, что далее по маршруту следования потока некоторое коммуникационное устройство применяет профилирование. Если профиль, задаваемый для формирования трафика, совпадает с профилем последующего профилирования, то это гарантирует отсутствие потерь трафика из-за отбрасывания избыточных пакетов.

Механизмы кондиционирования трафика могут либо поддерживаться каждым узлом сети, либо реализовываться только в пограничных устройствах. Последний вариант часто используют поставщики услуг, кондиционируя трафик своих клиентов.

Алгоритм ведра маркеров

Алгоритм ведра маркеров используется как для сглаживания, так и для профилирования трафика.

Он основан на сравнении потока пакетов с некоторым эталонным потоком. Эталонный поток представлен маркерами, заполняющими условное «ведро» маркеров (рис. 6.11).

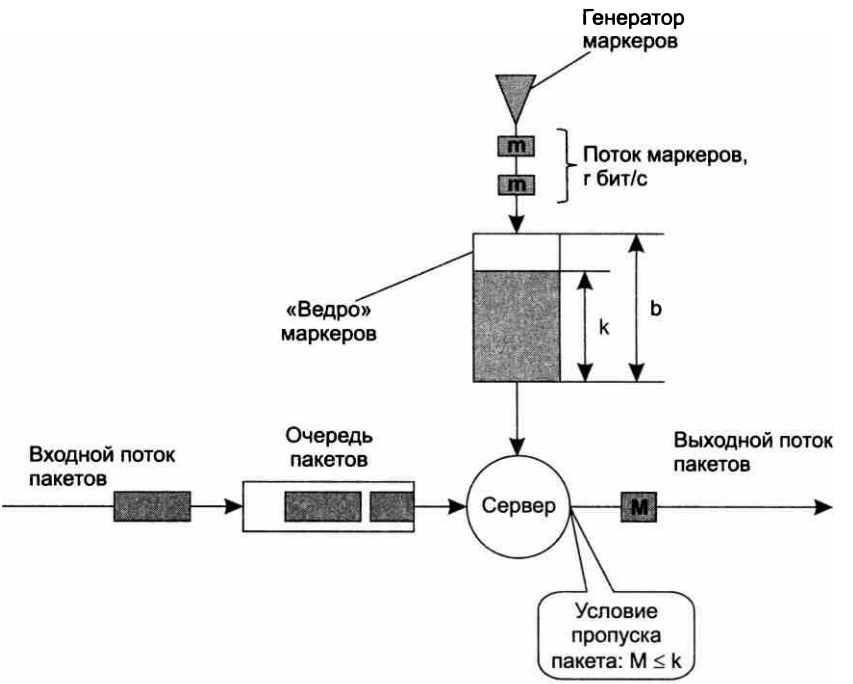


Рис. 6.11. Алгоритм ведра маркеров

Под маркером в данном случае понимается некий абстрактный объект, носитель «порции» информации, используемый для построения эталонного потока. Генератор маркеров периодически с постоянным интервалом w направляет очередной маркер в ведро с ограниченным объемом b байт. Все маркеры имеют одинаковый объем m байт, а генерация маркеров

происходит так, что ведро заполняется со скоростью r бит/с. Нетрудно подсчитать, что $r = 8m/w$. Эта скорость r и является максимальной средней скоростью для трафика пакетов, а объем ведра соответствует максимальному размеру пульсации потока пакетов. Если ведро заполняется маркерами «до краев» (то есть суммарный объем маркеров в ведре становится равным b), то поступление маркеров временно прекращается. Фактически ведро маркеров представляет собой счетчик, который наращивается на величину m каждые w секунд.

При применении алгоритма ведра маркеров профиль трафика определяется **средней скоростью** r и **объемом пульсации** b.

Сравнение эталонного и реального потоков выполняет сервер — абстрактное устройство, которое имеет два входа. Вход 1 связан с очередью пакетов, а вход 2 — с ведром маркеров. Сервер также имеет выход, на который он передает пакеты из входной очереди пакетов. Вход 1 сервера моделирует входной интерфейс маршрутизатора, а выход — выходной интерфейс.

> Пакет из входной очереди продвигается сервером на выход только в том случае, если к моменту его поступления на сервер ведро заполнено маркерами до уровня не ниже M байт, где M — объем пакета.

При продвижении пакета из ведра удаляются маркеры общим объемом в M байт (с точностью до размера одного маркера, то есть до m байт).

Если же ведро заполнено недостаточно, то пакет обрабатывается одним из двух описанных далее нестандартных способов, выбор которых зависит от цели применения алгоритма.

- Если алгоритм ведра маркеров применяется для *сглаживания* трафика, то пакет просто задерживается в очереди на некоторое дополнительное время, ожидая поступления в ведро нужного числа маркеров. Таким образом, даже если в результате пульсации в систему приходит большая группа пакетов, из очереди пакеты выходят более равномерно — в темпе, задаваемом генератором маркеров.
- Если же алгоритм ведра маркеров используется для *профилирования* трафика, то пакет отбрасывается как не соответствующий профилю. Более мягким решением может быть повторная маркировка пакета, понижающая его статус при дальнейшем обслуживании. Например, пакет может быть помечен особым признаком «удалять при необходимости», в результате чего при перегрузках маршрутизаторы будут отбрасывать такие пакеты в первую очередь. При дифференцированном обслуживании пакет может быть переведен в другой класс, который обслуживается с более низким качеством.

Алгоритм ведра маркеров допускает пульсацию трафика в определенных пределах. Пусть пропускная способность выходного интерфейса, который моделируется выходом сервера, равна R. Это значит, что сервер не может передавать данные на выход со скоростью, превышающей R бит/с. Можно показать, что на любом интервале времени t средняя скорость исходящего с сервера потока равна минимуму из двух величин: R и $r + b/t$. При больших значениях t скорость выходного потока стремится к r — это и говорит о том, что алгоритм обеспечивает желаемую среднюю скорость. В то же время в течение небольшого периода времени t пакеты могут выходить из сервера со скоростью, большей r. Если $r + b/t < R$, то они выходят из сервера со скоростью $r + b/t$, в противном случае интерфейс ограничивает эту скорость до величины R. Период времени t соответствует пульсации трафика. Эта ситуация наблюдается тогда, когда в течение некоторого времени пакеты не

поступали на сервер, так что ведро полностью заполнилось маркерами (то есть времени, большего, чем b/r). Если после этого на вход сервера поступит большая группа пакетов, следующих один за другим, то эти пакеты будут передаваться на выход со скоростью выходного интерфейса R также один за другим, без интервалов. Максимальное время такой пульсации составляет $b/(R - r)$ секунд, после чего обязательно наступит пауза, необходимая для наполнения опустевшего ведра. Объем пульсации составляет $Rb/(R - r)$ байт. Из приведенного соотношения видно, что алгоритм ведра маркеров начинает плохо работать, если средняя скорость r выбирается близкой к пропускной способности выходного интерфейса. В этом случае пульсация может продолжаться очень долго, что обесценивает алгоритм.

Обратная связь

Алгоритмы управления очередями и кондиционирования трафика не предотвращают перегрузок, а лишь некоторым «справедливым» образом в условиях дефицита перераспределяют ресурсы между различными потоками или классами трафика. Алгоритмы управления очередями относятся к механизмам **управления перегрузкой** (congestion management), которые начинают работать, когда сеть уже перегружена. Существует другой класс средств, которые носят название механизмов **предотвращения перегрузок** (congestion avoidance).

Этот механизм основан на использовании *обратной связи*, с помощью которой перегруженный узел сети, реагируя на перегрузку, просит предыдущие узлы, расположенные вдоль маршрута следования потока (или потоков, принадлежащих к одному классу), *временно* снизить скорость трафика. После того как перегрузка в данном узле исчезнет, он посылает другое сообщение, разрешающее повысить скорость передачи данных.

Существует несколько механизмов обратной связи (рис. 6.12). Они отличаются информацией, которая передается по обратной связи, а также тем, какой тип узла генерирует эту информацию и кто реагирует на эту информацию — конечный узел (компьютер) или промежуточный (коммутатор или маршрутизатор).

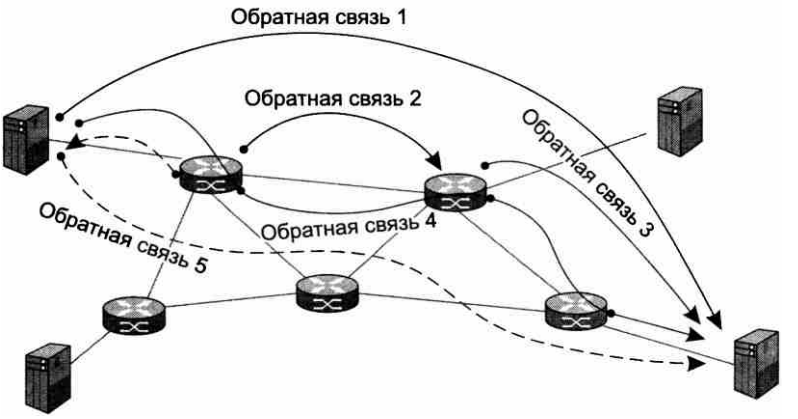


Рис. 6.12. Участники обратной связи

Обратная связь 1 организована между двумя конечными узлами сети. Этот вариант обеспечивает наиболее радикальное снижение нагрузки на сеть, так как только конечный узел может снизить скорость поступления информации в сеть. Назначение этого вида обратной связи — борьба с перегрузками узла назначения, а не с перегрузками промежуточных сетевых устройств, поэтому за ним закрепилось собственное название — **контроль потока**. Устройства сети не принимают участие в работе этого вида механизма обратной связи, они только передают соответствующие сообщения между конечными узлами. Несмотря на разные названия, в методах управления перегрузкой и контроля потока используются общие механизмы.

При организации обратной связи важно учитывать влияние, которое вносит задержка передачи информации по сети. Так, в высокоскоростных глобальных сетях за время, которое тратится на передачу сообщения о перегрузке узла назначения, узел-источник может успеть направить в сеть тысячи пакетов, так что перегрузка не будет ликвидирована вовремя. Из теории автоматического управления известно, что задержки в контуре обратной связи могут приводить ко многим нежелательным эффектам, прямо противоположным первоначальным целям. Например, в системе могут начаться колебательные процессы и она никогда не сможет прийти в равновесное состояние. Подобные явления наблюдались на ранней стадии развития Интернета, когда из-за несовершенства алгоритмов обратной связи и маршрутизации в нем возникали участки перегрузок, которые периодически перемещались по сети. Причина такой проблемы интуитивно понятна — задержка в контуре обратной связи приводит к тому, что регулирующий элемент получает устаревшую информацию о состоянии регулируемого элемента. Поэтому возможны ситуации, когда узел-источник начинает снижать скорость передачи информации, хотя в действительности очереди в узле-получателе уже нет, и наоборот, повышать скорость передачи информации в тот момент, когда узел-получатель начал испытывать перегрузку. Для борьбы с такими явлениями в контур обратной связи обычно вводится интегрирующий элемент, который на каждом шаге обрабатывает не только текущее сообщение обратной связи, но и несколько предыдущих сообщений, что позволяет учесть динамику изменения ситуации и реагировать адекватно.

Обратная связь 2 организована между двумя соседними коммутаторами. Коммутатор сообщает соседу, находящемуся выше по течению потока, что он испытывает перегрузку и его буфер заполнился до критической величины. Получив такое сообщение, сосед, расположенный выше по течению, должен снизить на некоторое время скорость передачи данных в направлении перегруженного коммутатора и тем самым решить проблему перегрузки. Это менее эффективное для сети в целом решение, так как поток будет продолжать течь от узла-источника с той же скоростью, что и раньше. Однако для коммутатора, который испытывает перегрузку, это является хорошим выходом, так как он получает время, чтобы разгрузить переполнившуюся очередь. Правда, проблема переносится в коммутатор, расположенный выше по течению, в котором теперь может возникнуть перегрузка, так как он начинает передавать данные из своего буфера с меньшей скоростью. Достоинством описанного метода является снижение задержки обратной связи, так как узлы являются соседями.

Обратная связь 3 организована между некоторым промежуточным коммутатором и узлом-источником; все остальные промежуточные коммутаторы, лежащие между этими двумя узлами, только передают сообщения обратной связи в направлении к узлу-источнику, никак на них не реагируя.

В *обратной связи* 4, как и в обратной связи 1, сообщение о перегрузке порождается узлом-получателем и передается узлу-источнику. Однако в данном случае каждый промежуточный коммутатор реагирует на это сообщение. Во-первых, он снижает скорость передачи данных в направлении узла назначения, во-вторых, он может изменить содержание сообщения. Например, если узел назначения просит снизить скорость до 300 Мбит/с, то промежуточный коммутатор может снизить эту величину до 200 Мбит/с, оценив состояние своего буфера. Кроме того, породить сообщение обратной связи может любой коммутатор сети, а не только узел назначения.

При описании различных вариантов организации обратной связи мы подразумевали, что сообщение о перегрузке идет в направлении, обратном направлению передачи пользовательской информации (собственно, поэтому этот механизм так и называется). Однако некоторые коммуникационные протоколы не предусматривают возможности генерации подобных сообщений промежуточными узлами. В таких условиях часто применяют искусственный прием — передачу сообщения о перегрузке узлу назначения, который преобразует его в сообщение обратной связи и отправляет в нужном направлении, то есть в направлении источника. Этот вариант показан на рисунке как *обратная связь* 5.

В применяемых сегодня методах обратной связи используются следующие основные типы сообщений:

- признак перегрузки;
- максимальная скорость передачи;
- максимальный объем данных;
- косвенные признаки.

Признак перегрузки не говорит о степени перегруженности сети или узла, он только фиксирует факт наличия перегрузки. Реакция узла, получившего такое сообщение, может быть разной. В некоторых протоколах узел обязан прекратить передачу информации в определенном направлении до тех пор, пока не будет получено другое сообщение обратной связи, разрешающее продолжение передачи. В других протоколах узел ведет себя адаптивно, он снижает скорость на некоторую величину и ожидает реакции сети. Если сообщения с признаком перегрузки продолжают поступать, то он продолжает снижение скорости.

Во втором типе сообщений указывается **максимальная скорость передачи**, то есть порог скорости, который должен соблюдать источник или промежуточный узел, расположенный выше по течению потока. В этом случае обязательно нужно учитывать время передачи сообщения по сети, чтобы исключить колебательные процессы в сети и обеспечить нужную скорость реакции на перегрузку. Поэтому в территориальных сетях такой способ обычно реализуется силами всех коммутаторов сети (обратная связь 4 в нашем примере).

Сообщение о **максимальном объеме данных** используется в широко применяемом в пакетных сетях алгоритме скользящего окна (подробнее о нем рассказывается в главе 16). Этот алгоритм позволяет не только обеспечивать надежную передачу данных, но и реализовать обратную связь для контроля потока между конечными узлами. Параметром, несущим информацию обратной связи, является «окно» — число, тесно связанное с текущим размером свободного пространства в буфере принимающего узла. Передающий узел может с любой скоростью передать объем информации, равный определенному для него окну. Но если этот лимит исчерпан, то передающий узел не имеет права передавать информацию,

пока не получит следующее окно. При перегрузках принимающий узел уменьшает размер окна, тем самым снижая нагрузку. Если эффект перегрузки исчезает, то принимающий узел увеличивает размер окна. Недостатком этого алгоритма является то, что он работает только в протоколах с установлением соединения.

В некоторых случаях передающий узел определяет, что принимающий узел (или узлы) испытывает перегрузку, по некоторым **косвенным признакам**, без получения сообщения обратной связи. Такими косвенными признаками могут быть факты потери пакетов. Примером протокола, использующего неявную информацию о перегрузках, является протокол TCP. Этот протокол с помощью явной информации обратной связи (о размере окна) осуществляет контроль потока, а с помощью неявной (потери пакетов, дубликаты квитанций) управляет перегрузкой.

Резервирование ресурсов

Рассмотренные методы поддержания качества обслуживания ориентированы в основном на борьбу с перегрузками или предотвращение их в пределах отдельного узла сети. Вместе с тем понятно, что для поддержания гарантированного уровня качества обслуживания некоторого потока пакетов необходимо скоординированное применение этих методов на всем пути следования потока через сеть.

Резервирование ресурсов — это координирующая процедура, которая настраивает все механизмы поддержания качества обслуживания вдоль следования потока таким образом, чтобы поток с некоторыми заданными характеристиками скорости был обслужен с заданными характеристиками QoS.

Основная идея процедуры резервирования ресурсов состоит в следующем. *Перед тем как* реальный поток будет направлен в сеть, каждому узлу сети вдоль маршрута его следования задается вопрос, может ли этот узел обслужить некоторый новый поток с заданными характеристиками QoS, если известны предельные характеристики скорости потока, такие как средняя и пиковая скорости? Каждый узел при ответе на этот вопрос должен оценить свои возможности, то есть проверить, достаточно ли у него свободных ресурсов, чтобы принять на обслуживание новый поток и обслуживать его качественно. При положительном ответе узел должен некоторым образом зарезервировать часть своих ресурсов для данного потока, чтобы при поступлении пакетов потока на входные интерфейсы использовать эти ресурсы для обслуживания поступающих пакетов с гарантированным уровнем качества.

В общем случае каждый узел самостоятельно решает, какие ресурсы он должен зарезервировать для обслуживания некоторого потока с заданным качеством. Как показывает практика, основным ресурсом, требуемым для качественного обслуживания пакетов, является пропускная способность интерфейса, через который пакеты потока покидают узел. Поэтому в дальнейшем мы будем, несколько упрощая действительное положение дел, употреблять формулировку «резервирование пропускной способности» вместо «резервирование ресурсов».

Однако что же означает резервирование пропускной способности в сетях с коммутацией пакетов? Мы сталкивались с этой концепцией только при рассмотрении принципов функ-

ционирования сетей с коммутацией каналов. Действительно, для сетей с коммутацией пакетов механизм резервирования пропускной способности не является принципиально необходимым, он имеет *вспомогательное* значение и используется только в тех случаях, когда требуется гарантированное обеспечение заданного качества обслуживания пакетов. Процедура резервирования здесь подобна аналогичной процедуре в сетях с коммутацией каналов: определенному потоку данных назначается определенная часть пропускной способности линии связи. Однако в сетях с коммутацией пакетов эта процедура является более гибкой, а именно — если отведенная пропускная способность в какой-то период времени недоиспользуется потоком, то она может быть передана другим потокам. Еще одним отличием резервирования в пакетных сетях является то обстоятельство, что резервирование может выполняться не только «из конца в конец», но и для каких-то отдельных узлов по маршруту потока.

Контроль допуска

Резервирование пропускной способности в пакетной сети «из конца в конец» начинается с операции, называемой **контролем допуска в сеть** (admission control) потока, который просит зарезервировать для своего обслуживания некоторую пропускную способность сети между ее двумя конечными узлами. Эта операция состоит в проверке наличия доступной (то есть незарезервированной для других потоков) пропускной способности на каждом из узлов сети на протяжении всего маршрута следования потока. Очевидно, что максимальная средняя скорость потока должна быть меньше, чем запрашиваемая пропускная способность, иначе поток будет обслужен с очень плохим качеством, даже несмотря на то, что ему была зарезервирована некоторая пропускная способность.

Если результат контроля допуска положителен в каждом узле (случай, показанный на рис. 6.13), то сетевые устройства запоминают факт резервирования, чтобы при появлении пакетов данного потока распознать их и выделить им зарезервированную пропускную способность. Кроме того, при успешном резервировании доступная для резервирования (в будущем) пропускная способность уменьшается на величину, зарезервированную за данным потоком.

Давайте теперь посмотрим, каким же образом выполняется собственно выделение пропускной способности потоку в моменты времени, когда его пакеты поступают на вход коммуникационного устройства $S2$, которое запомнило факт резервирования пропускной способности для потока $F1$ на выходном интерфейсе $P2$ (рис. 6.14).

Такое выделение можно обеспечить разными способами, в нашем примере это будет сделано с использованием взвешенных очередей. Пусть потоку $F1$ при резервировании было выделено 25 % пропускной способности интерфейса $P2$. Для простоты будем считать, что резервирование было выполнено только для потока $F1$, а для всех других потоков, которые проходят через выходной интерфейс $P2$, резервирования не производилось.

Для того чтобы добиться желаемого результата, достаточно организовать для выходного интерфейса две взвешенные очереди — очередь для потока $F1$ с весом 25 % и очередь «по умолчанию» для всех остальных потоков. Кроме того, необходимо активизировать *классификатор*, который будет проверять пакеты на всех входных интерфейсах устройства $S2$ (на рис. 6.14 показан только один входной интерфейс $P1$), отбирать пакеты потока $F1$

по заданным при резервировании признакам и направлять их в очередь для потока *F*1. В те периоды времени, когда скорость потока *F*1 окажется меньше зарезервированной пропускной способности в 25 %, неиспользованная ее часть будет потребляться потоками из очереди «по умолчанию» — в силу алгоритма работы взвешенных очередей. Зато в периоды, когда скорость потока *F*1 достигнет заявленного максимума потребления пропускной способности в 25 %, все остальные потоки будут довольствоваться оставшимися 75 %.

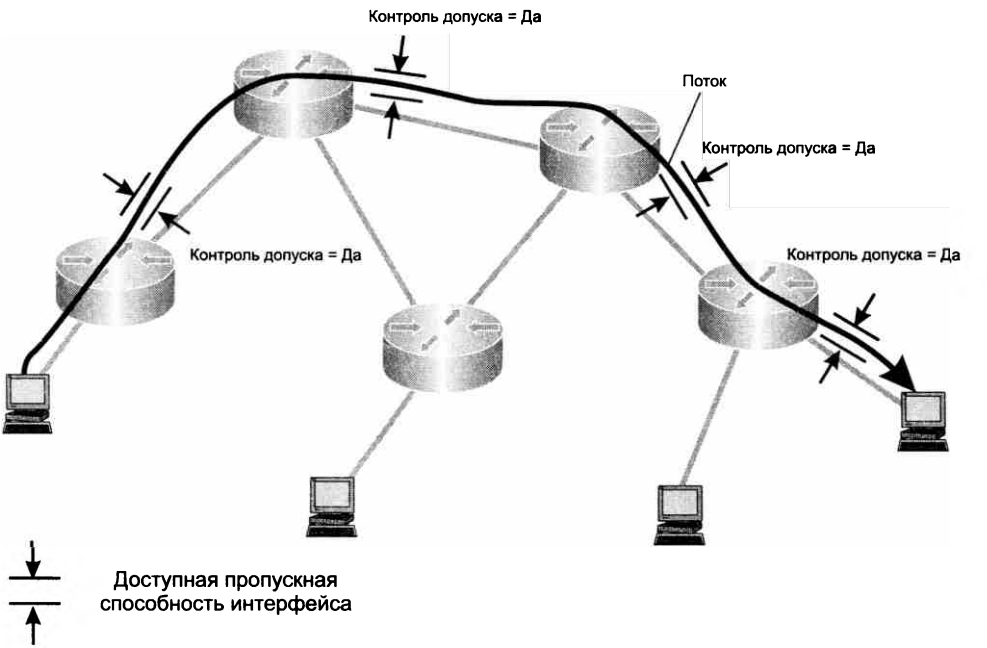


Рис. 6.13. Контроль допуска потока

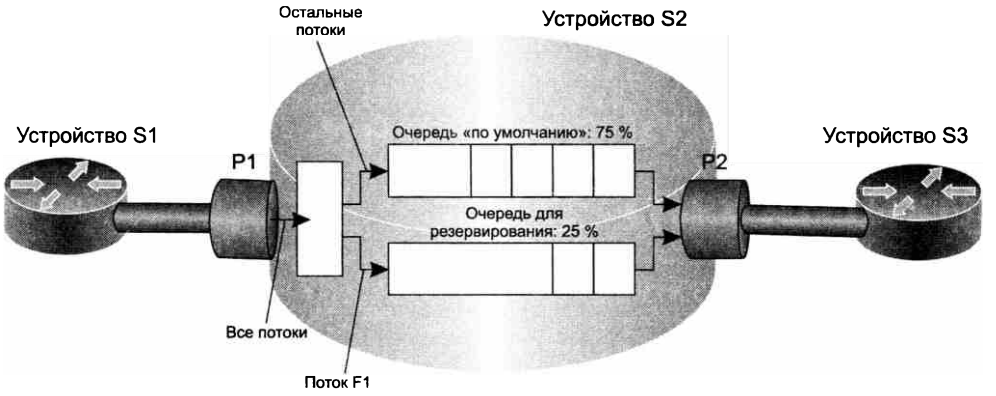


Рис. 6.14. Выделение зарезервированной пропускной способности

В описанном примере не задействован механизм профилирования трафика. При наличии отдельной взвешенной очереди для потока, зарезервировавшего пропускную способность, этот механизм не является обязательным, так как сам механизм взвешенных очередей ограничит пропускную способность потока в нужных пределах в периоды перегрузок, когда все взвешенные очереди заполняются полностью.

Использование взвешенных очередей — не единственный вариант резервирования пропускной способности в пакетных сетях. Для той же цели можно задействовать приоритетные очереди. Применение приоритетной очереди может быть не только возможным, но и необходимым, если потоку помимо определенного уровня пропускной способности требуется обеспечить минимально возможный уровень задержек пакетов.

При использовании приоритетной очереди профилирование необходимо всегда, так как приоритетный механизм не обеспечивает ограничения скорости потока, как это делает механизм взвешенного обслуживания.

Нужно подчеркнуть, что резервирование приводит к ожидаемым результатам только в тех случаях, когда реальная скорость потоков, для которых было выполнено резервирование, оказывается не выше, чем пропускная способность, запрошенная при резервировании и реализованная при конфигурировании сетевых устройств. В противном случае результаты могут оказаться даже хуже, чем при наличии единственной очереди «по умолчанию» и обслуживании «по возможности». Так, если скорость потока окажется выше, чем предел, учитываемый механизмом профилирования, то часть пакетов будет отброшена даже в том случае, если устройство не перегружено и могло бы отлично справиться с предложенным трафиком без применения механизмов QoS.

Обеспечение заданного уровня задержек

При описании процедуры резервирования пропускной способности мы сфокусировались на механизмах выделения пропускной способности некоторому потоку и оставили без внимания одну важную деталь: какую пропускную способность должен запрашивать поток, для того чтобы задержки его пакетов не превышали некоторой величины? Единственное соображение, которое было высказано по этому поводу, заключалось в том, что запрашиваемая пропускная способность должна быть выше, чем максимальная скорость потока, иначе некоторая часть пакетов просто может постоянно отбрасываться сетью, так что качество обслуживания окажется гарантированно низким.

Однако эта «деталь» на самом деле оборачивается сложной проблемой, так как мы не можем, например, сконфигурировать очередь приоритетного или взвешенного обслуживания так, чтобы она строго обеспечила какой-либо заранее заданный порог задержек и их вариации. Направление пакетов в приоритетную очередь только позволяет гарантировать, что задержки будут достаточно низкими — существенно ниже, чем у пакетов, которые обрабатываются в очереди по умолчанию. Мы также знаем, что при наличии взвешенных очередей задержки будут снижаться со снижением относительного коэффициента использования пропускной способности, отведенной очереди. Но это все качественные рассуждения, а вот количественно оценить значения задержек очень сложно.

Каким же образом поставщик услуг может выполнить свои обязательства перед клиентами? Очень «просто» — он должен постоянно *измерять фактические значения характеристик трафика в сети* и гарантировать пользователям сети величины задержек в соответ-

ствии с наблюдаемыми результатами. На практике обеспечить постоянный мониторинг задержек и потерь пакетов в сети оказывается совсем не просто — это требует установки в сети большого количества агентов-измерителей, хорошо синхронизированных друг с другом, а также программной системы регистрации и анализа измерительной информации. Поэтому операторы часто предпочитают давать качественное описание различных классов услуг, говоря, например, о минимальных задержках наивысшего класса обслуживания, но не давая количественных гарантий.

Инжиниринг трафика

При рассмотрении системы обеспечения качества обслуживания, основанной на резервировании, мы не стали затрагивать вопрос маршрутов следования потоков через сеть. Точнее, мы считали, что маршруты каким-то образом выбраны, причем этот выбор делается без учета требований QoS. И в условиях заданности маршрутов мы старались обеспечить прохождение по этим маршрутам такого набора потоков, для которого можно гарантировать соблюдение требований QoS.

Очевидно, что задачу обеспечения требований QoS можно решить более эффективно, если считать, что маршруты следования трафика не фиксированы, а также подлежат выбору. Это позволило бы сети обслуживать больше потоков с гарантиями QoS при тех же характеристиках самой сети, то есть пропускной способности каналов и производительности коммутаторов и маршрутизаторов.

Задачу выбора маршрутов для потоков (или классов) трафика с учетом соблюдения требований QoS решают методы **инжиниринга трафика** (Traffic Engineering). С помощью этих методов стремятся добиться еще одной цели — по возможности максимально и сбалансированно загрузить все ресурсы сети, чтобы сеть при заданном уровне качества обслуживания обладала как можно более высокой суммарной производительностью.

Методы инжиниринга трафика, как и другие рассмотренные ранее методы, основаны на резервировании ресурсов. То есть они не только позволяют найти рациональный маршрут для потока, но и резервируют для него пропускную способность ресурсов сети, находящихся вдоль этого маршрута.

Недостатки традиционных методов маршрутизации

Основным принципом работы протоколов маршрутизации в сетях с коммутацией пакетов вот уже долгое время является выбор маршрута на основе топологии сети без учета информации о ее текущей загрузке.

Для каждой пары «адрес источника — адрес назначения» такие протоколы выбирают единственный маршрут, не принимая во внимание информационные потоки, протекающие через сеть. В результате все потоки между парами конченых узлов сети идут по *кратчайшему* (в соответствии с некоторой метрикой) маршруту. Выбранный маршрут может быть более рациональным, например, если в расчет принимается номинальная пропускная способность каналов связи или вносимые ими задержки, или менее рациональным, если

учитывается только количество промежуточных маршрутизаторов между исходным и конечным узлами.

Классическим примером неэффективности такого подхода является так называемая «рыба» — сеть с топологией, приведенной на рис. 6.15. Несмотря на то что между коммутаторами *A* и *E* существуют два пути (верхний — через коммутатор *B*, и нижний — через коммутаторы *C* и *D*), весь трафик от коммутатора *A* к коммутатору *E* в соответствии с традиционными принципами маршрутизации направляется по верхнему пути. Только потому, что нижний путь немного (на один ретрансляционный участок) длиннее, чем верхний, он игнорируется, хотя мог бы работать «параллельно» с верхним путем.

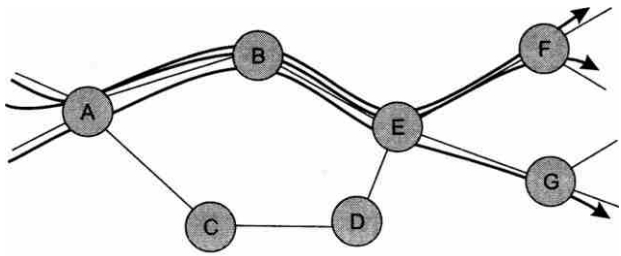


Рис. 6.15. Неэффективность кратчайших путей

Такой подход приводит к тому, что даже если кратчайший путь перегружен, пакеты все равно посылаются по этому пути. Налицо явная ущербность методов распределения ресурсов сети — одни ресурсы работают с перегрузкой, а другие не используются вовсе. Традиционные методы борьбы с перегрузками эту проблему решить не могут, нужны качественно иные механизмы.

Методы инжиниринга трафика

Исходными данными для методов инжиниринга трафика являются:

- характеристики передающей сети;
- сведения о предложенной нагрузке сети.

К *характеристикам передающей сети* относится ее топология, а также производительность составляющих ее коммутаторов и линий связи. Предполагается, что производительность процессора каждого коммутатора достаточна для обслуживания трафика всех его входных интерфейсов, даже если трафик поступает на интерфейс с максимально возможной скоростью, равной пропускной способности интерфейса. При таких условиях в качестве резервируемых ресурсов выступает пропускная способность линий связи между коммутаторами (рис. 6.16).

Сведения о предложенной нагрузке сети представляют собой информацию о потоках трафика, которые сеть должна передать между своими пограничными коммутаторами. Каждый поток характеризуется точкой входа в сеть, точкой выхода из сети и профилем трафика. Для получения оптимальных решений можно использовать детальное описание каждого потока, например учитывать величину возможной пульсации трафика. Однако поскольку количественно оценить их влияние на работу сети достаточно сложно, а влияние

этих параметров на характеристики QoS менее значимо, для нахождения субоптимального распределения путей прохождения потоков через сеть, как правило, учитываются только их средние скорости передачи данных (рис. 6.17).

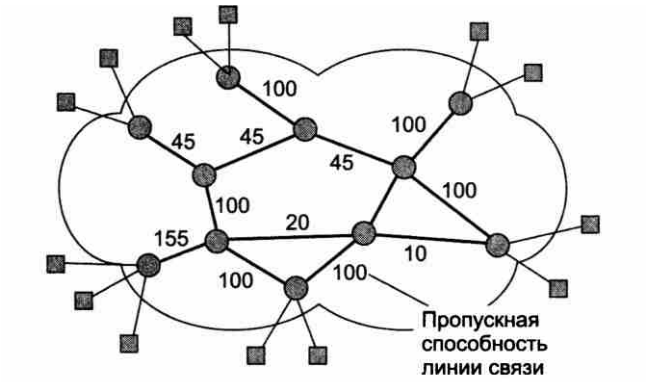


Рис. 6.16. Топология сети и производительность ее ресурсов

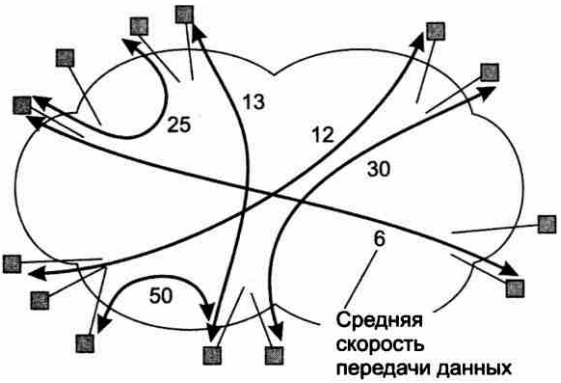


Рис. 6.17. Предложенная нагрузка

Методы инжиниринга трафика чаще применяют не к отдельным, а к *агрегированным* потокам, которые являются объединением нескольких потоков. Так как мы ищем общий маршрут для нескольких потоков, то агрегировать можно только потоки, имеющие общие точки входа в сеть и выхода из сети. Агрегированное задание потоков позволяет упростить задачу выбора путей, так как при индивидуальном рассмотрении каждого пользовательского потока промежуточные коммутаторы должны хранить слишком большие объемы информации, поскольку индивидуальных потоков может быть очень много. Необходимо, однако, подчеркнуть, что агрегирование отдельных потоков в один возможно только в том случае, когда все потоки, составляющие агрегированный поток, предъявляют одни и те же требования к качеству обслуживания. Далее в этом разделе мы будем для краткости пользоваться термином «поток» как для индивидуального потока, так и для агрегированного, поскольку принципы инжиниринга трафика от этого не меняются.

Задача инжиниринга трафика состоит в определении маршрутов прохождения потоков через сеть, то есть для каждого потока требуется найти точную последовательность промежуточных коммутаторов и их интерфейсов. При этом маршруты должны быть такими, чтобы все ресурсы сети были нагружены до максимально возможного уровня, а каждый поток получал требуемое качество обслуживания.

Максимальный уровень использования ресурсов выбирается таким образом, чтобы механизмы управления перегрузкой могли обеспечить требуемое качество обслуживания. Это означает, что для эластичного трафика максимальное значение выбирается не больше чем 0,9, а для чувствительного к задержкам трафика — не больше чем 0,5. Так как обычно резервирование производится не для всех потоков, нужно оставить часть пропускной способности для свободного использования. Поэтому приведенные максимальные значения обычно уменьшают до 0,75 и 0,25 соответственно. Для упрощения рассуждений мы будем считать далее, что в сети передается один вид трафика, а потом покажем, как обобщить решение задачи инжиниринга для случая трафика нескольких типов.

Существуют различные формальные математические постановки задачи инжиниринга трафика. Мы здесь ограничимся наиболее простой из них, тем более что сегодня она чаще всего используется на практике: решением задачи инжиниринга трафика является такой набор маршрутов для заданного множества потоков трафика, для которого все значения коэффициентов использования ресурсов вдоль маршрута следования каждого потока не превышают некоторого заданного порога K_{max}.

Решение задачи инжиниринга трафика можно искать по-разному. Во-первых, можно искать его заблаговременно, *в фоновом режиме*. Для этого нужно знать исходные данные: топологию и производительность сети, а также предложенную нагрузку. После этого задачу рационального распределения путей следования трафика при фиксированных точках входа и выхода, а также заданном уровне максимального значения коэффициента использования ресурса можно передать некоторой программе, которая, например, путем направленного перебора вариантов найдет точные маршруты для каждого потока с указанием всех промежуточных коммутаторов.

Во-вторых, задачу инжиниринга трафика можно решать *в оперативном режиме*, поручив ее самим коммутаторам сети. Для этого используются модифицированные стандартные протоколы маршрутизации. Модификация протоколов маршрутизации состоит в том, что они сообщают друг другу не только топологическую информацию, но и текущее значение свободной пропускной способности для каждого ресурса.

После того как решение найдено, нужно его реализовать, то есть отразить в таблицах маршрутизации. На этом этапе может возникнуть проблема — в том случае, если мы хотим проложить эти маршруты в дейтаграммной сети. Дело в том, что таблицы маршрутизации в них учитывают только адреса назначения пакетов. Коммутаторы и маршрутизаторы таких сетей (например, IP-сетей) не работают с потоками, для них поток в явном виде не существует, каждый пакет при его продвижении является независимой единицей коммутации. Можно сказать, что таблицы продвижения этих сетей отражают только топологию сети (направления продвижения к определенным адресам назначения).

Поэтому привнесение методов резервирования в дейтаграммные сети происходит с большими трудностями. В протоколах резервирования, чтобы определить поток для дейтаграммного маршрутизатора, помимо адреса назначения используется некоторый дополнительный набор признаков. При этом понятие потока привлекается только на этапе

резервирования, а при продвижении пакетов по-прежнему работает традиционная для этого типа сетей схема, учитывающая лишь адрес назначения.

Теперь представим ситуацию, когда между двумя конечными узлами имеется несколько потоков, которые требуется направить по разным маршрутам. Такое решение было принято исходя из баланса загрузки сети, то есть в результате инжиниринга трафика. Дейтаграммный коммутатор или маршрутизатор не имеет возможности реализовать наше решение, потому что для всех этих потоков у него в таблице продвижения есть только одна запись, соответствующая общему адресу назначения пакетов этих потоков. В таких условиях есть только одно чрезвычайно трудно реализуемое на практике решение — изменение логики работы коммутаторов и маршрутизаторов.

В связи с этим методы инжиниринга трафика сегодня используются только в сетях с виртуальными каналами, для которых не составляет труда реализовать найденное решение для группы потоков. Каждому потоку (или группе потоков с одинаковыми маршрутами) выделяется виртуальный канал, который прокладывается в соответствии с выбранным маршрутом. Методы инжиниринга трафика успешно применялись в сетях ATM и Frame Relay до тех пор, пока эти технологии не прекратили свое существование. Сегодня задачи инжиниринга трафика решаются в сетях IP поверх MPLS, так как MPLS использует технику виртуальных каналов для продвижения пакетов.

Работа в недогруженном режиме

Как мы уже отмечали, самым простым способом обеспечения требований QoS для всех потоков является работа сети в недогруженном режиме, или с избыточной пропускной способностью.

Говорят, что *сеть имеет избыточную пропускную способность*, когда все части сети в любой момент времени обладают такой пропускной способностью, которой достаточно, чтобы обслужить все потоки трафика, протекающего в это время через сеть, с удовлетворительными характеристиками производительности и надежности. Другими словами, ни одно из сетевых устройств такой сети никогда не подвергается перегрузкам, которые могли бы привести к значительным задержкам или потерям пакетов из-за переполнения очередей пакетов (конечно, это не исключает случаев потерь сетью пакетов по другим причинам, не связанным с перегрузкой сети, например из-за искажений сигналов в линиях связи либо отказов сетевых узлов или линий связи).

Заметим, что приведенное определение сети с избыточной пропускной способностью намеренно упрощено, чтобы донести суть идеи. Более аккуратное определение должно было бы учитывать случайный характер протекающих в сети процессов и оперировать статистическими определениями событий. Например, оговаривать, что такие события, как длительные задержки или потери пакетов из-за переполнения очередей в сети с избыточной пропускной способностью, случаются настолько редко, что ими можно пренебречь.

Простота обеспечения требований QoS за счет работы сети в недогруженном режиме является главным достоинством этого подхода — он требует только увеличения пропускной способности линий связи и соответственно производительности коммуникационных устройств сети. Никаких дополнительных усилий по исследованию характеристик потоков

сети и конфигурированию дополнительных очередей и механизмов кондиционирования трафика, как в случае применения методов QoS, здесь не требуется.

Чтобы быть уверенным, что сеть обладает достаточной пропускной способностью для качественной передачи трафика, необходим постоянный мониторинг временны́х характеристик (задержек и их вариаций) процессов передачи пакетов сетью. А в том случае, когда результаты мониторинга начинают стабильно показывать ухудшение характеристик качества обслуживания, необходимо проводить очередную модернизацию сети и увеличивать пропускную способность линий связи и коммуникационных устройств.

Однако мониторинг задержек и их вариаций является тонкой и трудоемкой работой. Обычно операторы, которые хотят поддерживать свою сеть в недогруженном состоянии и за счет этого обеспечивать высокое качество обслуживания, решают более простую задачу — они осуществляют мониторинг уровня трафика в линиях связи сети, то есть *измеряют коэффициент использования пропускной способности линий связи*. При этом линия связи считается недогруженной, если ее коэффициент использования постоянно не превосходит некоторый достаточно низкий уровень, например 20–30 %. Имея такие значения измерений, можно считать, что линия в среднем не испытывает перегрузок, а значит, задержки пакетов будут низкими.

Выводы

Методы обеспечения качества обслуживания занимают сегодня важное место в семействе технологий сетей с коммутацией пакетов, так как без их применения сложно обеспечить качественную работу современных мультимедийных приложений, таких как IP-телефония, видео- и радиовещание, интерактивное дистанционное обучение и т. п.

Характеристики QoS отражают отрицательные последствия пребывания пакетов в очередях, которые проявляются в снижении скорости передачи, задержках пакетов и их потерях.

Существуют различные типы трафика, отличающиеся чувствительностью к задержкам и потерям пакетов. Наиболее грубая классификация трафика разделяет его на два класса: трафик реального времени (чувствительный к задержкам) и эластичный трафик (нечувствительный к задержкам в широких пределах).

Методы QoS основаны на перераспределении имеющейся пропускной способности линий связи между трафиком различного типа в соответствии с требованиями приложений.

Приоритетные и взвешенные очереди являются основным инструментом выделения пропускной способности определенным потокам пакетов.

Механизм профилирования позволяет контролировать скорость потока пакетов и ограничивать ее в соответствии с заранее заданным уровнем.

Обратная связь является одним из механизмов QoS; она позволяет временно снизить скорость поступления пакетов в сеть для ликвидации перегрузки в узле сети.

Резервирование пропускной способности «из конца в конец» позволяет добиться гарантированного качества обслуживания потока пакетов. Резервирование основано на процедуре контроля допуска потока в сеть, в ходе которой проверяется наличие доступной пропускной способности для обслуживания потока вдоль маршрута его следования.

Методы инжиниринга трафика состоят в выборе рациональных маршрутов прохождения потоков через сеть. Выбор маршрутов обеспечивает максимизацию загрузки ресурсов сети при одновременном соблюдении необходимых гарантий в отношении параметров качества обслуживания трафика.

Недогруженная сеть может обеспечить качественное обслуживание трафика всех типов без применения методов QoS; однако для того, чтобы убедиться, что сеть действительно недогружена, требуется постоянно проводить мониторинг уровней загрузки линий связи сети, выполняя измерения с достаточно высокой частотой.

Контрольные вопросы

1. Возникают ли очереди в сетях с коммутацией каналов?
2. К каким нежелательным последствиям может привести приоритетное обслуживание?
3. Объясните причину возможного возникновения очередей даже при невысокой средней загрузке коммутаторов или маршрутизаторов сети с коммутацией пакетов?
4. Какой тип обслуживания целесообразно применить, если нужно обеспечить различную минимальную гарантированную способность трем классам трафика?
5. Какой параметр трафика меняется при инжиниринге трафика?

Часть II

Технологии физического уровня

- Глава 7. Линии связи
- Глава 8. Кодирование и мультиплексирование данных
- Глава 9. Беспроводная передача данных
- Глава 10. Первичные сети

Физической основой любой компьютерной (и телекоммуникационной) сети являются линии связи. Без таких линий коммутаторы не могли бы обмениваться пакетами и компьютеры оставались бы изолированными устройствами.

После изучения принципов построения компьютерных сетей в воображении читателя могла возникнуть достаточно простая картина компьютерной сети — компьютеры и коммутаторы, соединенные друг с другом отрезками кабеля. Однако при более детальном рассмотрении компьютерной сети все оказывается сложнее, чем это казалось при изучении модели OSI.

Дело в том, что специально выделенные кабели используются для соединения сетевых устройств только на небольших расстояниях, то есть в локальных сетях. При построении сетей WAN и MAN такой подход крайне расточителен из-за высокой стоимости протяженных линий связи. К тому же на их прокладку необходимо получать разрешение. Поэтому гораздо чаще для связи коммутаторов в сетях WAN и MAN применяются существующие первичные территориальные сети с коммутацией каналов. В этом случае в сети с коммутацией каналов создается составной канал, который выполняет те же функции, что и отрезок кабеля, — обеспечивает физическое двухточечное соединение. Конечно, составной канал представляет собой гораздо более сложную техническую систему, чем кабель, но для компьютерной сети эти сложности прозрачны. Первичные сети специально строятся для создания канальной инфраструктуры, поэтому их каналы более эффективны по соотношению цена/пропускная способность. Сегодня в распоряжении проектировщика компьютерной сети имеются каналы первичных сетей для широкого диапазона скоростей — от 64 Кбит/с до 10 Гбит/с.

Несмотря на различия в физической и технической природе линий связи, их можно описать с помощью единого набора характеристик. Важнейшими характеристиками любой линии связи при передаче дискретной информации являются полоса пропускания, измеряемая в герцах (Гц), и пропускная способность, измеряемая в битах в секунду (бит/с). Пропускная способность представляет собой скорость битового потока, передаваемого линией связи. Пропускная способность зависит от полосы пропускания линии и способа кодирования дискретной информации.

Все большую популярность приобретают беспроводные каналы. Они являются единственным типом каналов, обеспечивающих мобильность пользователей компьютерной сети.

ГЛАВА 7 Линии связи

Классификация линий связи

Первичные сети, линии и каналы связи

При описании технической системы, которая передает информацию между узлами сети, в литературе можно встретить несколько названий: *линия связи*, *составной канал*, *канал*, *звено*. Часто эти термины используются как синонимы, и во многих случаях это не вызывает проблем. В то же время есть и специфика в их употреблении.

- **Звено** (link) — это сегмент, обеспечивающий передачу данных между двумя соседними узлами сети. То есть звено не содержит промежуточных устройств коммутации и мультиплексирования.
- **Каналом** (channel) чаще всего обозначают часть пропускной способности звена, используемую независимо при коммутации. Например, звено первичной сети может состоять из 30 каналов, каждый из которых обладает пропускной способностью 64 Кбит/с.
- **Составной канал** (circuit) — это путь между двумя конечными узлами сети. Составной канал образуется отдельными каналами промежуточных звеньев и внутренними соединениями в коммутаторах. Часто определение «составной» опускается, и термином «канал» называют как составной канал, так и канал между соседними узлами, то есть в пределах звена.
- **Линия связи** может использоваться как синоним для любого из трех остальных терминов.

Не стоит относиться к путанице в терминологии очень строго. Особенно это относится к различиям в терминологии традиционной телефонии и более новой области — компьютерных сетей. Процесс конвергенции только усугубил проблему терминологии, так как многие механизмы этих сетей стали общими, но сохранили за собой по паре (иногда и больше) названий, пришедших из каждой области.

Кроме того, существуют объективные причины для неоднозначного понимания терминов. На рис. 7.1 показаны два варианта линии связи. В первом случае линия состоит из сегмента кабеля длиной несколько десятков метров и представляет собой звено (рис. 7.1, *а*). Во втором случае линия связи представляет собой составной канал, проложенный в сети с коммутацией каналов (рис. 7.1, *б*). Такой сетью может быть **первичная сеть** или телефонная сеть (ее элементы поясняются ниже в разделе «Аппаратура передачи данных»).

Однако для компьютерной сети эта линия представляет собой звено, так как соединяет два соседних узла (например, два маршрутизатора), и вся коммутационная промежуточная аппаратура является прозрачной для этих узлов. Повод для взаимного непонимания на уровне терминов компьютерных специалистов и специалистов первичных сетей здесь очевиден.

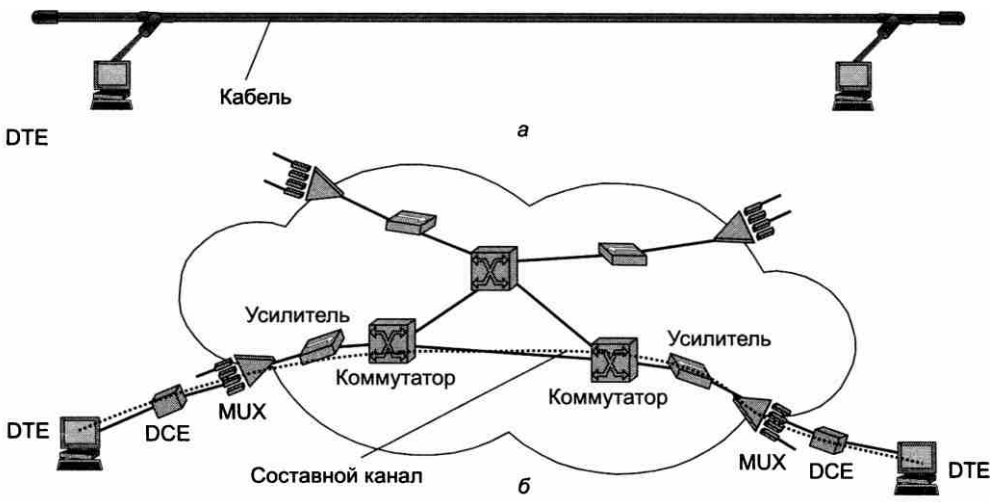


Рис. 7.1. Состав линии связи

Первичные сети специально создаются для того, чтобы предоставлять услуги каналов передачи данных для компьютерных и телефонных сетей, про которые в таких случаях говорят, что они работают «поверх» первичных сетей и являются **наложенными сетями**.

Физическая среда передачи данных

Линии связи отличаются также физической средой, используемой для передачи информации.

Физическая среда передачи данных может представлять собой набор проводников, по которым передаются сигналы. На основе таких проводников строятся проводные (воздушные) или кабельные линии связи (рис. 7.2). В качестве среды также используется земная атмосфера или космическое пространство, через которое распространяются информационные сигналы. В первом случае говорят о *проводной среде*, а во втором — о *беспроводной*.

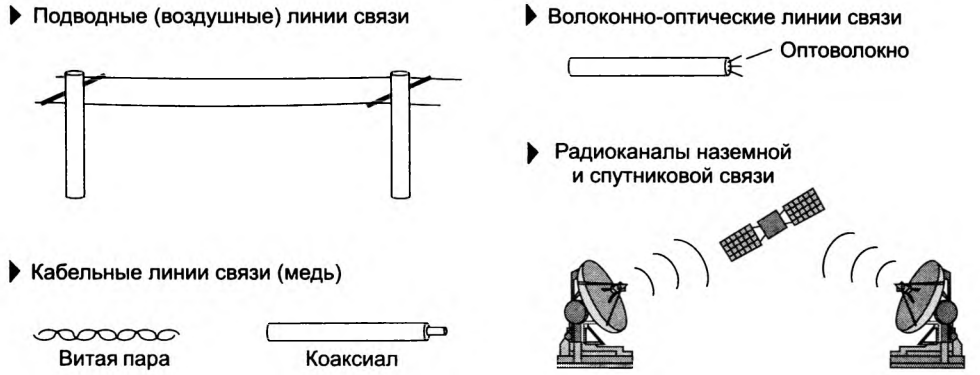


Рис. 7.2. Типы сред передачи данных

В современных телекоммуникационных системах информация передается с помощью электрического тока или напряжения, радиосигналов или световых сигналов — все эти физические процессы являются колебаниями электромагнитного поля различной частоты.

Проводные (воздушные) линии связи представляют собой провода без каких-либо изолирующих или экранирующих оплеток, проложенные между столбами и висящие в воздухе. Еще в недалеком прошлом такие линии связи были основными для передачи телефонных и телеграфных сигналов. Сегодня проводные линии связи быстро вытесняются кабельными. Но кое-где они все еще сохранились и при отсутствии других возможностей продолжают использоваться, в частности и для передачи компьютерных данных. Скоростные качества и помехозащищенность этих линий оставляют желать много лучшего.

Кабельные линии имеют достаточно сложную конструкцию. Кабель состоит из проводников, заключенных в несколько слоев изоляции: электрической, электромагнитной, механической и, возможно, климатической. Кроме того, кабель может быть оснащен разъемами, позволяющими быстро выполнять присоединение к нему различного оборудования. В компьютерных (и телекоммуникационных) сетях применяются три основных типа кабеля: кабели на основе скрученных пар медных проводов — **неэкранированная витая пара** (Unshielded Twisted Pair, UTP) и **экранированная витая пара** (Shielded Twisted Pair, STP), **коаксиальные кабели** с медной жилой, **волоконно-оптические кабели**. Первые два типа кабелей называют также **медными кабелями**.

Радиоканалы наземной и спутниковой связи образуются с помощью передатчика и приемника радиоволн. Существует большое разнообразие типов радиоканалов, отличающихся как используемым частотным диапазоном, так и дальностью канала. **Диапазоны широковещательного радио** (длинных, средних и коротких волн), называемые также **АМ-диапазонами**, или диапазонами амплитудной модуляции (Amplitude Modulation, AM), обеспечивают дальнюю связь, но при невысокой скорости передачи данных. Более скоростными являются каналы, использующие **диапазоны очень высоких частот** (Very High Frequency, VHF), в которых применяется частотная модуляция (Frequency Modulation, FM). Для передачи данных также используются **диапазоны ультравысоких частот** (Ultra High Frequency, UHF), называемые еще **диапазонами микроволн** (свыше 300 МГц). При такой частоте сигналы уже не отражаются ионосферой Земли и для устойчивой связи требуется наличие прямой видимости между передатчиком и приемником. Поэтому указанные частоты используются в спутниковых или радиорелейных каналах либо в таких локальных или мобильных сетях, в которых это условие выполняется.

В компьютерных сетях сегодня применяются практически все описанные типы физических сред передачи данных. Хорошие возможности предоставляют волоконно-оптические кабели, обладающие широкой полосой пропускания и низкой чувствительностью к помехам. На них сегодня строятся как магистрали крупных территориальных и городских сетей, так и высокоскоростные локальные сети. Популярной средой является также витая пара, которая характеризуется отличным отношением качества к стоимости, а также простотой монтажа. Беспроводные каналы используются чаще всего в тех случаях, когда кабельные линии связи применить нельзя, например при прохождении канала через малонаселенную местность или же для связи с мобильными пользователями сети. Обеспечение мобильности затронуло в первую очередь телефонные сети, компьютерные сети в этом отношении пока отстают. Тем не менее построение компьютерных сетей на основе беспроводных технологий, например Radio Ethernet, считается сегодня одним из самых перспективных направлений телекоммуникаций. Линии связи на основе беспроводной среды изучаются в главе 9.

Аппаратура передачи данных

Как показано на рис. 7.1, линии связи состоят не только из среды передачи, но и аппаратуры. Даже в том случае, когда линия связи не проходит через первичную сеть, а основана на кабеле, в ее состав входит аппаратура передачи данных.

Аппаратура передачи данных (Data Circuit Equipment, DCE) в компьютерных сетях непосредственно присоединяет компьютеры или коммутаторы к линиям связи и является, таким образом, пограничным оборудованием. Традиционно аппаратуру передачи данных включают в состав линии связи. Примерами DCE являются **модемы** (для телефонных линий), **терминальные адаптеры сетей ISDN**, **устройства для подключения к цифровым каналам** первичных сетей DSU/CSU (Data Service Unit/Circuit Service Unit).

DCE работает на физическом уровне модели OSI, отвечая за передачу информации в физическую среду (в линию) и прием из нее сигналов нужной формы, мощности и частоты. Аппаратура пользователя линии связи, вырабатывающая данные для передачи по линии связи и подключаемая непосредственно к аппаратуре передачи данных, носит обобщенное название **оконечное оборудование данных** (Data Terminal Equipment, DTE). Примером DTE могут служить компьютеры, коммутаторы и маршрутизаторы. Эту аппаратуру не включают в состав линии связи.

ПРИМЕЧАНИЕ

Разделение оборудования на DCE и DTE в локальных сетях является достаточно условным. Например, адаптер локальной сети можно считать как принадлежностью компьютера, то есть оборудованием DTE, так и составной частью канала связи, то есть аппаратурой DCE. Точнее, одна часть сетевого адаптера выполняет функции DTE, а его другая, оконечная часть, непосредственно принимающая и передающая сигналы, относится к DCE.

Для подключения DCE-устройств к DTE-устройствам (то есть к компьютерам или коммутаторам/маршрутизаторам) существует несколько *стандартных интерфейсов*[1]. Работают эти устройства на коротких расстояниях друг от друга, как правило, несколько метров.

Промежуточная аппаратура обычно используется на линиях связи большой протяженности. Она решает две основные задачи:

- улучшение качества сигнала;
- создание постоянного составного канала связи между двумя абонентами сети.

В локальных сетях промежуточная аппаратура может совсем не использоваться, если протяженность физической среды — кабелей или радиоэфира — позволяет одному сетевому адаптеру принимать сигналы непосредственно от другого сетевого адаптера без дополнительного усиления. В противном случае применяется промежуточная аппаратура, роль которой здесь играют устройства типа **повторителей** и **концентраторов**.

В *глобальных сетях* необходимо обеспечить качественную передачу сигналов на расстояния в сотни и тысячи километров. Поэтому без **усилителей** (повышающих мощность сигналов) и **регенераторов** (наряду с повышением мощности восстанавливающих форму импульс-

[1] Интерфейсы DTE-DCE описываются стандартами серии V CCITT, а также стандартами EIA серии RS (Recommended Standards — рекомендуемые стандарты). Две линии стандартов во многом дублируют друг друга. Наиболее популярными стандартами являются RS-232, RS-530, V.35 и HSSI.

ных сигналов, исказившихся при передаче на большое расстояние), установленных через определенные расстояния, построить территориальную линию связи невозможно.

В первичных сетях помимо упомянутого оборудования, обеспечивающего качественную передачу сигналов, необходима промежуточная коммутационная аппаратура — **мультиплексоры** (MUX), **демультиплексоры** и **коммутаторы**. Эта аппаратура создает между двумя абонентами сети постоянный составной канал из отрезков физической среды — кабелей с усилителями.

В зависимости от типа промежуточной аппаратуры все линии связи делятся на аналоговые и цифровые. В **аналоговых линиях** промежуточная аппаратура предназначена для усиления аналоговых сигналов, то есть сигналов, которые имеют непрерывный диапазон значений. Такие линии связи традиционно применялись в телефонных сетях с целью связи телефонных коммутаторов между собой. Для создания высокоскоростных каналов, которые мультиплексируют несколько низкоскоростных аналоговых абонентских каналов, при аналоговом подходе обычно используется техника *частотного мультиплексирования* (Frequency Division Multiplexing, FDM).

В **цифровых линиях** связи передаваемые сигналы имеют конечное число состояний. Как правило, элементарный сигнал, то есть сигнал, передаваемый за один такт работы передающей аппаратуры, имеет 2, 3 или 4 состояния, которые в линиях связи воспроизводятся импульсами или потенциалами прямоугольной формы. С помощью таких сигналов передаются как компьютерные данные, так и оцифрованные речь и изображение (именно благодаря одинаковому способу представления информации современными компьютерными, телефонными и телевизионными сетями стало возможным появление общих для всех первичных сетей). В цифровых линиях связи используется специальная промежуточная аппаратура — регенераторы, которые улучшают форму импульсов и восстанавливают период их следования. Промежуточная аппаратура мультиплексирования и коммутации первичных сетей работает по принципу *временного мультиплексирования каналов* (Time Division Multiplexing, TDM).

Характеристики линий связи

Спектральный анализ сигналов на линиях связи

Важная роль при определении параметров линий связи отводится спектральному разложению передаваемого по этой линии сигнала. Из теории гармонического анализа известно, что *любой периодический процесс можно представить в виде суммы синусоидальных колебаний различных частот и различных амплитуд* (рис. 7.3).

Каждая составляющая синусоида называется также **гармоникой**, а набор всех гармоник называют **спектральным разложением**, или **спектром**, исходного сигнала. Под **шириной спектра сигнала** понимается разность между максимальной и минимальной частотами того набора синусоид, которые в сумме дают исходный сигнал.

Непериодические сигналы можно представить в виде интеграла синусоидальных сигналов с непрерывным спектром частот. В частности, спектральное разложение идеального импульса (единичной мощности и нулевой длительности) имеет составляющие всего спектра частот, от $-\infty$ до $+\infty$ (рис. 7.4).

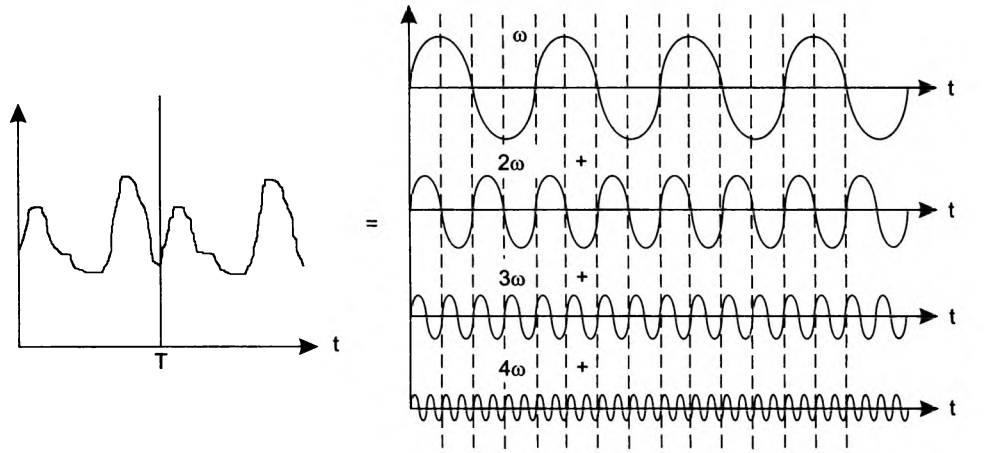


Рис. 7.3. Представление периодического сигнала суммой синусоид

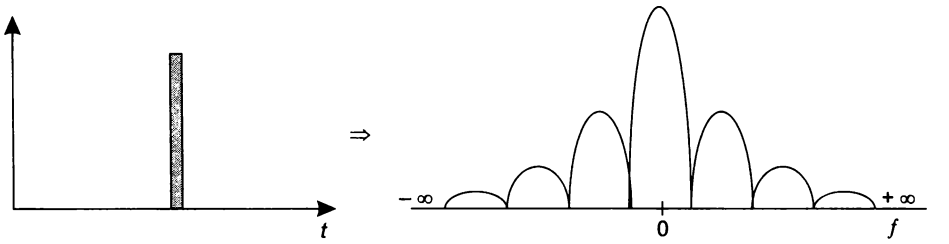


Рис. 7.4. Спектральное разложение идеального импульса

Техника нахождения спектра любого исходного сигнала хорошо известна. Для некоторых сигналов, которые описываются аналитически (например, для последовательности прямоугольных импульсов одинаковой длительности и амплитуды), спектр легко вычисляется на основании **формул Фурье**.

Для сигналов произвольной формы, встречающихся на практике, спектр можно найти с помощью специальных приборов — спектральных анализаторов, которые измеряют спектр реального сигнала и отображают амплитуды составляющих гармоник на экране, распечатывают их на принтере или передают для обработки и хранения в компьютер.

Искажение передающей линией связи синусоиды какой-либо частоты приводит в конечном счете к искажению амплитуды и формы передаваемого сигнала любого вида. Искажения формы проявляются в том случае, когда синусоиды различных частот искажаются неодинаково. Если это аналоговый сигнал, передающий речь, то изменяется тембр голоса за счет искажения обертонов — боковых частот. При передаче импульсных сигналов, характерных для компьютерных сетей, искажаются низкочастотные и высокочастотные гармоники, в результате фронты импульсов теряют свою прямоугольную форму (рис. 7.5) и сигналы могут плохо распознаваться на приемном конце линии.

Передаваемые сигналы искажаются из-за несовершенства линий связи. Для электрических сигналов идеальная передающая среда, не вносящая никаких помех в передаваемый

сигнал, должна по меньшей мере иметь нулевые значения сопротивления, емкости и индуктивности. Однако на практике медные провода, например, всегда представляют собой некоторую распределенную по длине комбинацию активного сопротивления, емкостной и индуктивной нагрузок (рис. 7.6). В результате синусоиды различных частот передаются этими линиями по-разному.

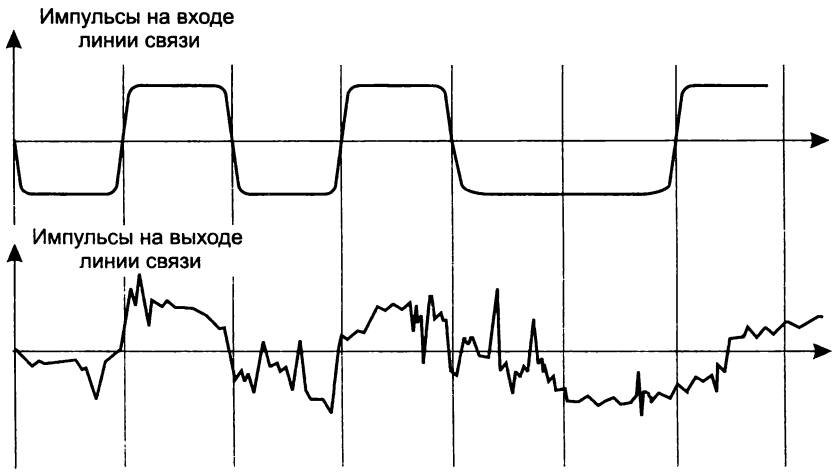


Рис. 7.5. Искажение импульсов в линии связи

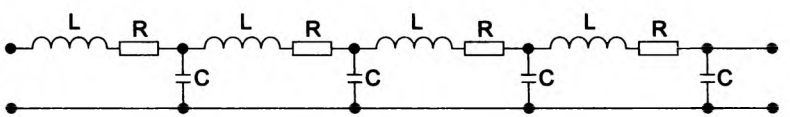


Рис. 7.6. Представление медной линии как распределенной индуктивно-емкостной нагрузки

Помимо искажений сигналов, возникающих из-за неидеальных физических параметров линии связи, существуют и **внешние помехи**, которые вносят свой вклад в искажение формы сигналов на выходе линии. Эти помехи создаются различными электрическими двигателями, электронными устройствами, атмосферными явлениями и т. д. Несмотря на защитные меры, предпринимаемые разработчиками кабелей, и наличие усилительной и коммутирующей аппаратуры, полностью компенсировать влияние внешних помех не удается. Помимо внешних помех в кабеле существуют и **внутренние помехи** — так называемые **наводки** одной пары проводников на другую. В результате сигналы на выходе линии связи могут иметь искаженную форму (как это и показано на рис. 7.5).

Затухание и волновое сопротивление

Степень искажения синусоидальных сигналов линиями связи оценивается такими характеристиками, как затухание и полоса пропускания.

Затухание показывает, насколько уменьшается мощность эталонного синусоидального сигнала на выходе линии связи по отношению к мощности сигнала на входе этой линии. Затухание (A) обычно измеряется в децибелах (дБ) и вычисляется по формуле

$$A = 10 \lg P_{out}/P_{in}.$$

Здесь P_{out} — мощность сигнала на выходе линии, P_{in} — мощность сигнала на входе линии. Так как затухание зависит от длины линии связи, то в качестве характеристики линии связи используется так называемое **погонное затухание**, то есть затухание на линии связи определенной длины. Для кабелей локальных сетей в качестве такой длины обычно используют 100 м, так как это значение является максимальной длиной кабеля для многих технологий LAN. Для территориальных линий связи погонное затухание измеряют для расстояния в 1 км.

Обычно затуханием характеризуют пассивные участки линии связи, состоящие из кабелей и кроссовых секций, без усилителей и регенераторов. Так как мощность выходного сигнала кабеля без промежуточных усилителей меньше, чем мощность входного, затухание кабеля всегда является *отрицательной величиной*.

Степень затухания мощности синусоидального сигнала зависит от частоты синусоиды, и эта зависимость также характеризует линию связи (рис. 7.7).

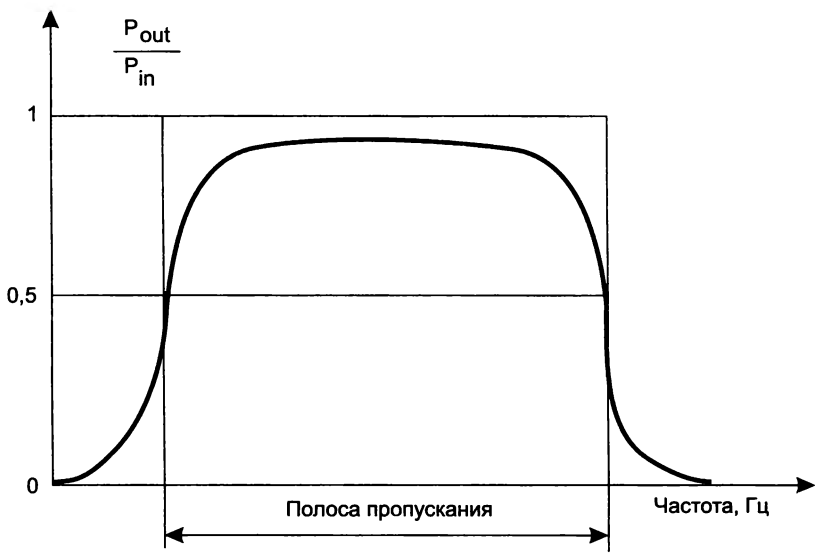


Рис. 7.7. Зависимость затухания от частоты

Чаще всего при описании параметров линии связи приводятся значения затухания всего *для нескольких значений частот*. Это объясняется, с одной стороны, стремлением упростить измерения при проверке качества линии. С другой стороны, на практике часто заранее известна основная частота передаваемого сигнала, то есть та частота, гармоника которой имеет наибольшие амплитуду и мощность. Поэтому достаточно знать затухание на этой частоте, чтобы приблизительно оценить искажения передаваемых по линии сигналов.

ВНИМАНИЕ

Как уже отмечалось, затухание всегда имеет отрицательное значение, однако знак «минус» часто опускают, из-за чего иногда возникает путаница. Совершенно корректно утверждение, что качество линии связи тем выше, чем больше (с учетом знака) затухание. Если же игнорировать знак, то есть иметь в виду абсолютное значение затухания, то у более качественной линии затухание меньше. Приведем пример. Для внутренней проводки в зданиях используется кабель на витой паре категории 5. Этот кабель, на котором работают практически все технологии локальных сетей, характеризуется затуханием, превышающем –23,6 дБ для частоты 100 МГц при длине кабеля 100 м. Более качественный кабель категории 6 имеет на частоте 100 МГц затухание больше чем –20,6 дБ. Получаем, что – 20,6 > –23,6, но 20,6 < 23,6.

На рис. 7.8 показаны типовые зависимости затухания от частоты для кабелей на неэкранированной витой паре категорий 5 и 6.

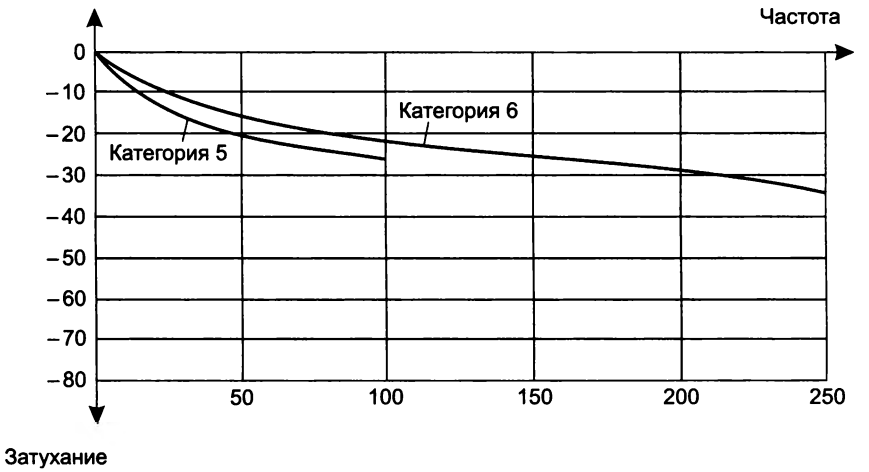


Рис. 7.8. Затухание неэкранированного кабеля на витой паре

Оптический кабель имеет существенно меньшие (по абсолютной величине) величины затухания, обычно в диапазоне от –0,2 до –3 дБ при длине кабеля в 1000 м, а значит, является более качественным, чем кабель на витой паре. Практически для всех оптических волокон типична сложная зависимость затухания от длины волны, которая имеет три так называемых **окна прозрачности**. На рис. 7.9 показана характерная зависимость затухания для оптического волокна. Из рисунка видно, что область эффективного использования современных волокон ограничена волнами длин 850 нм, 1300 нм и 1550 нм (соответственно частотами 35 ТГц, 23 ТГц и 19,4 ТГц). Окно 1550 нм обеспечивает наименьшие потери, а значит, максимальную дальность при фиксированной мощности передатчика и фиксированной чувствительности приемника.

В качестве характеристики мощности сигнала используются абсолютный и относительный уровни мощности. **Абсолютный уровень мощности** измеряется в ваттах, **относительный уровень мощности**, как и затухание, измеряется в децибелах.

Существует также и другая абсолютная единица измерения мощности — так называемая **опорная мощность**, измеряемая в децибелах на милливатт (дБм).

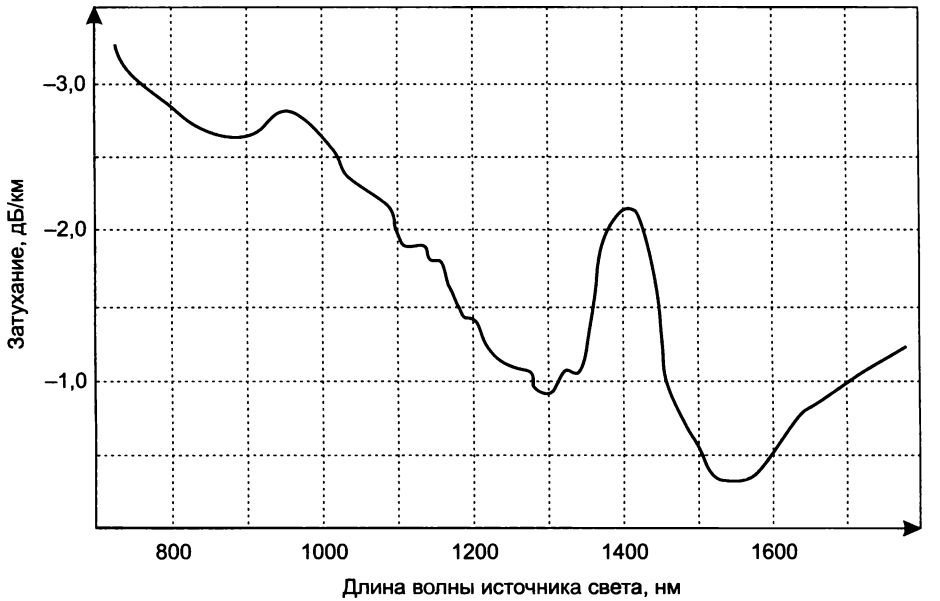


Рис. 7.9. Окна прозрачности оптического волокна

При определении опорной мощности также используется логарифм отношения мощностей, но значение мощности, к которой выполняется отношение, *фиксируется*. Опорный уровень мощности, а к нему относится измеряемая мощность, принимается равным 1 мВт, что и отражается в названии этой единицы мощности.

Опорная мощность p вычисляется по формуле

$$p = 10 \lg P/1\text{мВт}\ [\text{дБм}].$$

Здесь p — абсолютная мощность сигнала в милливаттах.

Несмотря на использование отношения в определении опорной мощности, эта единица измерения является *абсолютной*, а не относительной, так как однозначно преобразует абсолютную мощность сигнала в ваттах в некоторое значение, которое никак не зависит от значения мощности другого сигнала, как это имеет место при определении децибела. Так, нетрудно вычислить соответствие некоторых значений мощности сигнала, выраженных в ваттах и дБм:

1 мВ = 0 дБм

10 мВ = 10 дБм

1 В = 30 дБм

100 кВ = 80 дБм

Опорные значения мощности удобно использовать при *расчетах энергетического бюджета линий связи*.

Пример

Пусть требуется определить минимальную опорную мощность x (дБм) передатчика, достаточную для того, чтобы на выходе линии опорная мощность сигнала была не ниже некото-

рого порогового значения y (дБм). Затухание линии известно и равно A. Пусть X и Y — это абсолютные значения мощности сигнала, заданные в милливаттах на входе и выходе линии соответственно.

По определению $A = 10 \lg X/Y$. Используя свойства логарифмов, имеем:

$$A = 10 \lg X/Y = 10\lg(X/1)/(Y/1) = 10 \lg X/1 \text{ мВт} - 10 \lg Y/1 \text{ мВт}.$$

Заметим, что два последних члена уравнения по определению являются опорными значениями мощности сигналов на выходе и входе, поэтому приходим к простому соотношению $A = x - y$, где x — опорная мощность входного сигнала, а y — опорная мощность выходного сигнала.

Из последнего соотношения следует, что минимальная требуемая мощность передатчика может быть определена как сумма затухания и мощности сигнала на выходе: $x = A + y$.

Предельная простота расчета стала возможной благодаря тому, что в качестве исходных данных были взяты опорные значения мощности входного и выходного сигналов. Конечно, можно было бы использовать и значения мощностей, заданные в ваттах, но при этом пришлось бы заниматься такими операциями, как возведение 10 в дробную степень, что более громоздко.

Использованная в примере величина y называется **порогом чувствительности приемника** и представляет собой минимальную опорную мощность сигнала на входе приемника, при которой он способен корректно распознавать дискретную информацию, содержащуюся в сигнале. Очевидно, что для нормальной работы линии связи необходимо, чтобы минимальная опорная мощность сигнала передатчика, даже ослабленная затуханием линии связи, превосходила порог чувствительности приемника: $x - A > y$. Проверка этого условия и является сутью расчета энергетического бюджета линии.

Важным параметром медной линии связи является ее **волновое сопротивление**, представляющее собой полное (комплексное) сопротивление, которое встречает электромагнитная волна определенной частоты при распространении вдоль однородной цепи. Волновое сопротивление измеряется в омах и зависит от таких параметров линии связи, как активное сопротивление, погонная индуктивность и погонная емкость, а также от частоты самого сигнала. Выходное сопротивление передатчика должно быть согласовано с волновым сопротивлением линии, иначе затухание сигнала будет чрезмерно большим.

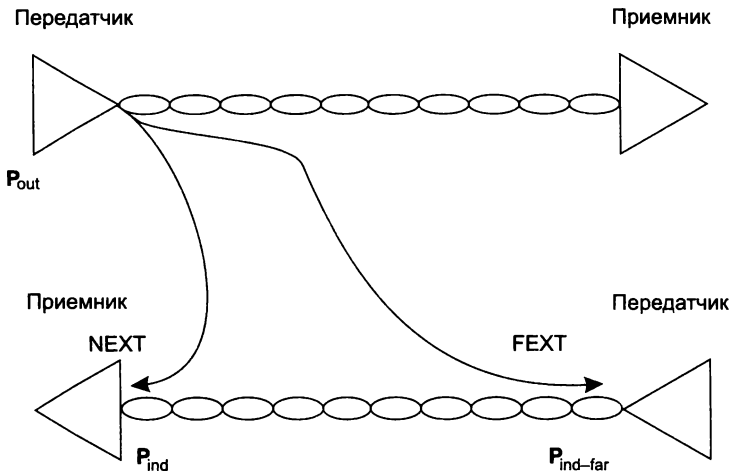
Помехоустойчивость и достоверность

Помехоустойчивость линии, как и следует из названия, определяет способность линии противостоять влиянию помех, создаваемых во внешней среде или на внутренних проводниках самого кабеля. Помехоустойчивость линии зависит от типа используемой физической среды, а также от средств экранирования и подавления помех самой линии. Наименее помехоустойчивыми являются радиолинии, хорошей устойчивостью обладают кабельные линии и отличной — волоконно-оптические линии, малочувствительные к внешнему электромагнитному излучению. Обычно для уменьшения помех, создаваемых внешними электромагнитными полями, проводники экранируют и/или скручивают.

Электрическая и магнитная связь — это параметры медного кабеля, также являющиеся результатом помех. **Электрическая связь** определяется отношением наведенного тока в подверженной влиянию цепи к напряжению, действующему во влияющей цепи. **Магнитная связь** — это отношение электродвижущей силы, наведенной в подверженной влиянию

цепи, к току во влияющей цепи. Результатом электрической и магнитной связи являются **наведенные сигналы** (наводки) в цепи, подверженной вилянию. Существует несколько различных параметров, характеризующих устойчивость кабеля к наводкам.

Перекрестные наводки на ближнем конце (Near End Cross Talk, NEXT) определяют устойчивость кабеля в том случае, когда наводка образуется в результате действия сигнала, генерируемого передатчиком, подключенным к одной из соседних пар на том же конце кабеля, на котором работает подключенный к подверженной влиянию паре приемник (рис. 7.10). Показатель NEXT, выраженный в децибелах, равен 10 lg P_{out}/P_{ind}, где P_{out} — мощность выходного сигнала, P_{ind} — мощность наведенного сигнала.



$P_{ind-far}$ – мощность наведенного сигнала на дальнем конце кабеля

Рис. 7.10. Переходное затухание

Чем больше значение NEXT, тем лучше кабель. Так, для витой пары категории 5 показатель NEXT должен быть больше 27 дБ на частоте 100 МГц.

Перекрестные наводки на дальнем конце (Far End Cross Talk, FEXT) позволяют оценить устойчивость кабеля к наводкам для случая, когда передатчик и приемник подключены к разным концам кабеля. Очевидно, что этот показатель должен быть лучше, чем NEXT, так как до дальнего конца кабеля сигнал приходит ослабленный затуханием каждой пары.

Показатели NEXT и FEXT обычно применяются к кабелю, состоящему из нескольких витых пар, так как в этом случае взаимные наводки одной пары на другую могут достигать значительных величин. Для одинарного коаксиального кабеля (то есть состоящего из одной экранированной жилы) этот показатель не имеет смысла, а для двойного коаксиального кабеля он также не применяется вследствие высокой степени защищенности каждой жилы. Оптические волокна тоже не создают сколько-нибудь заметных взаимных помех.

Достоверность передачи данных характеризует вероятность искажения каждого передаваемого бита данных. Иногда этот же показатель называют **интенсивностью битовых ошибок** (Bit Error Rate, BER). Величина BER для линий связи без дополнительных средств

защиты от ошибок (например, самокорректирующихся кодов или протоколов с повторной передачей искаженных кадров) составляет, как правило, 10^{-4}–10^{-6}, в оптоволоконных линиях связи — 10^{-9}. Например, значение достоверности передачи данных в 10^{-4} говорит о том, что в среднем из 10 000 бит искажается значение одного бита.

Полоса пропускания и пропускная способность

Полоса пропускания — это непрерывный диапазон частот, для которого затухание не превышает некоторый заранее заданный предел. То есть полоса пропускания определяет диапазон частот синусоидального сигнала, при которых этот сигнал передается по линии связи без значительных искажений.

Часто граничными частотами считаются частоты, на которых мощность выходного сигнала уменьшается в два раза по отношению к входному, что соответствует затуханию в –3 дБ. Как мы увидим далее, *ширина* полосы пропускания в наибольшей степени влияет на максимально возможную скорость передачи информации по линии связи. Полоса пропускания зависит от типа линии и ее протяженности. На рис. 7.11 показаны полосы пропускания линий связи различных типов, а также частотные диапазоны, наиболее часто используемые в технике связи.

Пропускная способность линии характеризует максимально возможную скорость передачи данных, которая может быть достигнута на этой линии. Особенностью пропускной способности является то, что, с одной стороны, эта характеристика зависит от параметров *физической среды*, а с другой — определяется *способом передачи данных*. Следовательно, нельзя говорить о пропускной способности линии связи до того, как для нее определен протокол физического уровня.

Например, если для цифровой линии определен протокол физического уровня, задающий фиксированную битовую скорость передачи данных, то для нее известна и пропускная способность — например, 2 Мбит/с, 100 Мбит/ с, 1 Гбит/с и т. п.

В тех же случаях, когда только предстоит выбрать, какой из множества существующих протоколов использовать на данной линии, очень важными являются остальные характеристики линии, такие как полоса пропускания, перекрестные наводки, помехоустойчивость и другие.

Пропускная способность, как и скорость передачи данных, измеряется в битах в секунду (бит/с), а также в производных единицах, таких как килобиты в секунду (Кбит/с) и т. д.

ВНИМАНИЕ

Пропускная способность линий связи и коммуникационного сетевого оборудования традиционно измеряется в битах в секунду, а не в байтах в секунду. Это связано с тем, что данные в сетях передаются последовательно, то есть побитно, а не параллельно, байтами, как это происходит между устройствами внутри компьютера. Такие единицы измерения, как килобит, мегабит или гигабит, в сетевых технологиях строго соответствуют степеням десяти (то есть килобит — это 1000 бит, а мегабит — это 1 000 000 бит), как это принято во всех отраслях науки и техники, а не близким к этим числам степеням двойки, как это принято в программировании, где приставка «кило» равна $2^{10} = 1024$, а «мега» — $2^{20} = 1\,048\,576$.

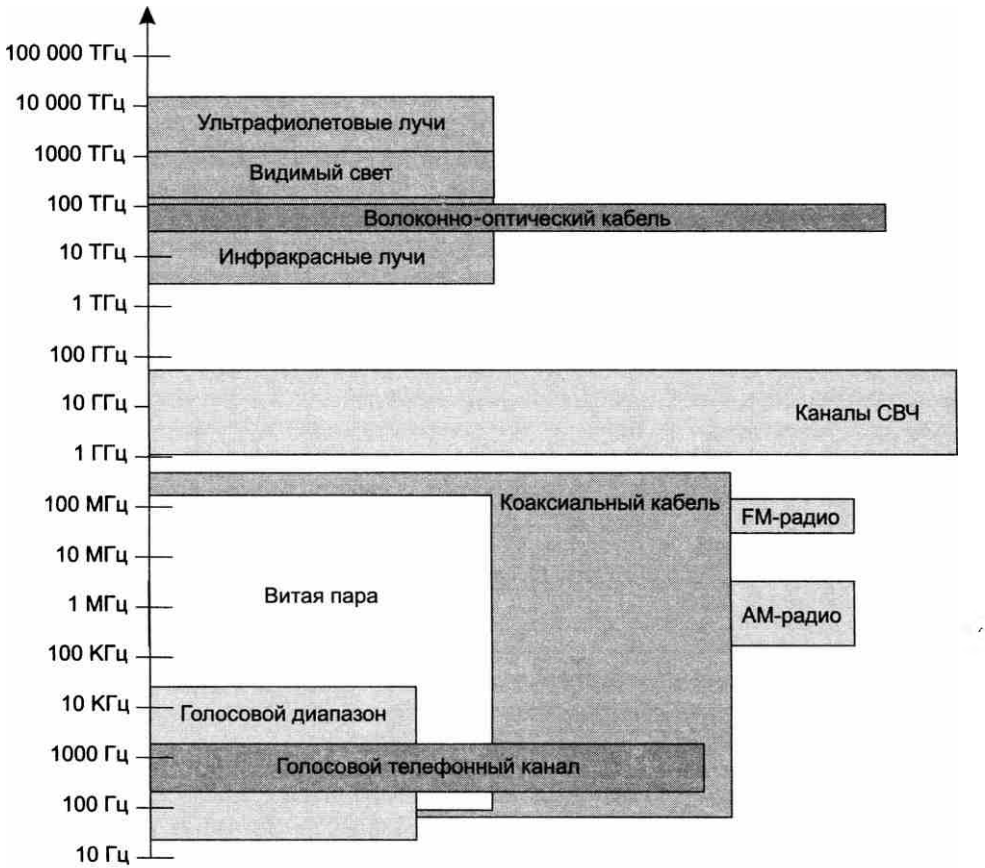


Рис. 7.11. Полосы пропускания линий связи и популярные частотные диапазоны

Пропускная способность линии связи зависит не только от ее характеристик, таких как затухание и полоса пропускания, но и от спектра передаваемых сигналов. Если значимые гармоники сигнала (то есть те гармоники, амплитуды которых вносят основной вклад в результирующий сигнал) попадают в полосу пропускания линии, то такой сигнал будет хорошо передаваться данной линией связи и приемник сможет правильно распознать информацию, отправленную по линии передатчиком (рис. 7.12, *а*). Если же значимые гармоники выходят за границы полосы пропускания линии связи, то сигнал начнет значительно искажаться и приемник будет ошибаться при распознавании информации (рис. 7.12, *б*).

Биты и боды

Выбор способа представления дискретной информации в виде сигналов, подаваемых на линию связи, называется **физическим**, или **линейным**, **кодированием**. От выбранного способа кодирования зависит спектр сигналов и соответственно пропускная способность линии.

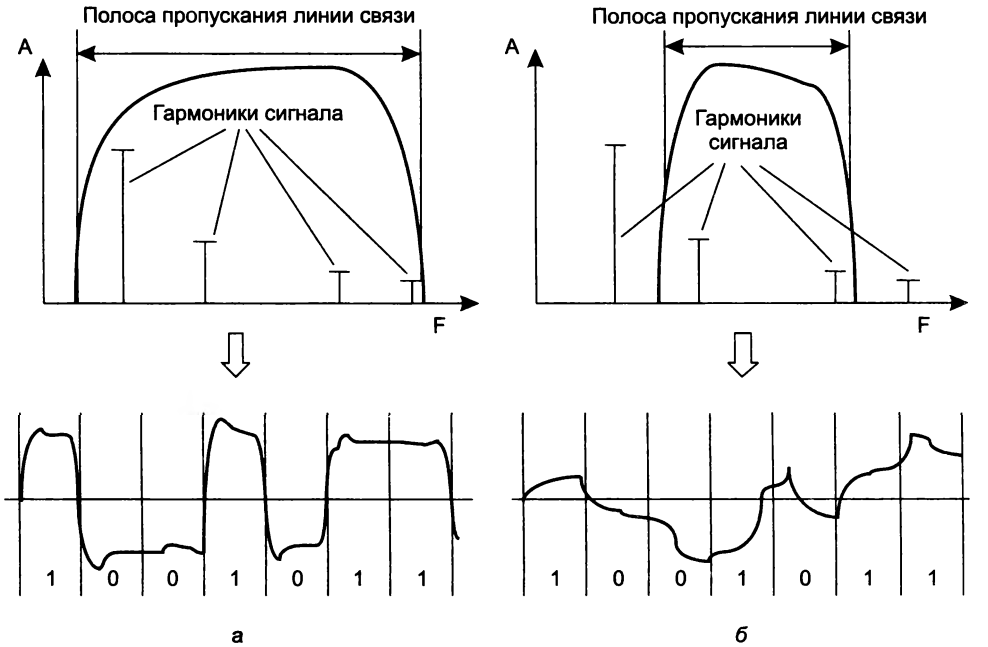


Рис. 7.12. Соответствие между полосой пропускания линии связи и спектром сигнала

Таким образом, для одного способа кодирования линия может обладать одной пропускной способностью, а для другого — другой. Например, витая пара категории 5 может передавать данные с пропускной способностью 100 Мбит/с при способе кодирования стандарта физического уровня 100Base-T и 1 Гбит/с при способе кодирования стандарта 1000Base-T.

ВНИМАНИЕ

В соответствии с основным постулатом теории информации любое различимое непредсказуемое изменение принимаемого сигнала несет в себе информацию. Отсюда следует, что синусоида, у которой амплитуда, фаза и частота остаются неизменными, информации не несет, так как изменение сигнала хотя и происходит, но является абсолютно предсказуемым. Аналогично не несут в себе информации импульсы на тактовой шине компьютера, так как их изменения тоже постоянны во времени. А вот импульсы на шине данных предсказать заранее нельзя, это и делает их информационными, они переносят информацию между отдельными блоками или устройствами компьютера.

В большинстве способов кодирования используется изменение какого-либо параметра периодического сигнала — частоты, амплитуды и фазы синусоиды или же знака потенциала последовательности импульсов. Периодический сигнал, параметры которого подвергаются изменениям, называют **несущим сигналом**, а его частоту, если сигнал синусоидальный, — **несущей частотой**. Процесс изменения параметров несущего сигнала в соответствии с передаваемой информацией называется **модуляцией**.

Если сигнал изменяется так, что можно различить только два его состояния, то любое его изменение будет соответствовать наименьшей единице информации — биту. Если же сиг-

нал может иметь более двух различимых состояний, то любое его изменение будет нести *несколько битов информации.*

Передача дискретной информации в телекоммуникационных сетях осуществляется тактировано, то есть изменение сигнала происходит через фиксированный интервал времени, называемый **тактом**. Приемник информации считает, что в начале каждого такта на его вход поступает новая информация. При этом независимо от того, повторяет ли сигнал состояние предыдущего такта или же он имеет состояние, отличное от предыдущего, приемник получает новую информацию от передатчика. Например, если такт равен 0,3 с, а сигнал имеет два состояния и 1 кодируется потенциалом 5 вольт, то присутствие на входе приемника сигнала величиной 5 вольт в течение 3 секунд означает получение информации, представленной двоичным числом 1111111111.

Количество изменений информационного параметра несущего периодического сигнала в секунду измеряется в **бодах**. 1 бод равен одному изменению информационного параметра в секунду. Например, если такт передачи информации равен 0,1 секунды, то сигнал изменяется со скоростью 10 бод. Таким образом, скорость в бодах целиком определяется длительностью такта.

Информационная скорость измеряется в битах в секунду и в общем случае *не совпадает* со скоростью в бодах. Она может быть как выше, так и ниже скорости изменения информационного параметра, измеряемого в *бодах*. Это соотношение зависит от числа состояний сигнала. Например, если сигнал имеет более двух различимых состояний, то при равных тактах и соответствующем методе кодирования информационная скорость в битах в секунду может быть *выше*, чем скорость изменения информационного сигнала в бодах.

Пусть информационными параметрами являются фаза и амплитуда синусоиды, причем различаются 4 состояния фазы в 0, 90, 180 и 270° и два значения амплитуды сигнала — тогда информационный сигнал может иметь 8 различимых состояний. Это означает, что любое состояние этого сигнала несет 3 бита информации. В этом случае модем, работающий со скоростью 2400 бод (меняющий информационный сигнал 2400 раз в секунду), передает информацию со скоростью 7200 бит/с, так как при одном изменении сигнала передается 3 бита информации.

Если сигнал имеет два состояния (то есть несет информацию в 1 бит), то информационная скорость обычно совпадает с количеством бодов. Однако может наблюдаться и обратная картина, когда информационная скорость оказывается *ниже* скорости изменения информационного сигнала в бодах. Это происходит в тех случаях, когда для надежного распознавания приемником пользовательской информации каждый бит в последовательности кодируется несколькими изменениями информационного параметра несущего сигнала. Например, при кодировании единичного значения бита импульсом положительной полярности, а нулевого значения бита импульсом отрицательной полярности физический сигнал дважды изменяет свое состояние при передаче каждого бита. При таком кодировании скорость линии в битах в секунду в два раза ниже, чем в бодах.

Чем выше частота несущего периодического сигнала, тем выше может быть частота модуляции и тем выше может быть пропускная способность линии связи.

Однако с увеличением частоты периодического несущего сигнала увеличивается и ширина спектра этого сигнала.

Линия передает этот спектр синусоид с теми искажениями, которые определяются ее полосой пропускания. Чем больше несоответствие между полосой пропускания линии и ши-

риной спектра передаваемых информационных сигналов, тем больше сигналы искажаются и тем вероятнее ошибки в распознавании информации принимающей стороной, а значит, возможная скорость передачи информации оказывается меньше.

Соотношение полосы пропускания и пропускной способности

Связь между полосой пропускания линии и ее пропускной способностью вне зависимости от принятого способа физического кодирования установил *Клод Шеннон*:

$$C = F \log_2 (1 + P_с/P_ш).$$

Здесь C — пропускная способность линии в битах в секунду, F — ширина полосы пропускания линии в герцах, $P_с$ — мощность сигнала, $P_ш$ — мощность шума.

Из этого соотношения следует, что теоретического предела пропускной способности линии с фиксированной полосой пропускания не существует. Однако на практике такой предел имеется. Действительно, повысить пропускную способность линии можно за счет увеличения мощности передатчика или же уменьшения мощности шума (помех) в линии связи. Обе эти составляющие поддаются изменению с большим трудом. Повышение мощности передатчика ведет к значительному увеличению его габаритов и стоимости. Снижение уровня шума требует применения специальных кабелей с хорошими защитными экранами, что весьма дорого, а также снижения шума в передатчике и промежуточной аппаратуре, чего достичь весьма непросто. К тому же влияние мощностей полезного сигнала и шума на пропускную способность ограничено логарифмической зависимостью, которая растет далеко не так быстро, как прямо пропорциональная. Так, при достаточно типичном исходном значении отношения мощности сигнала к мощности шума, равном 100, повышение мощности передатчика в два раза даст только 15 % увеличения пропускной способности линии.

Близким по сути к формуле Шеннона является другое соотношение, полученное *Найквистом*, которое также определяет *максимально возможную пропускную способность линии связи, но без учета шума в линии*:

$$C = 2F \log_2 M.$$

Здесь M — количество различимых состояний информационного параметра.

Если сигнал имеет два различимых состояния, то максимально возможная пропускная способность равна удвоенному значению ширины полосы пропускания линии связи (рис. 7.13, *а*). Если же в передатчике используется более двух устойчивых состояний сигнала для кодирования данных, то максимально возможная пропускная способность линии повышается, так как за один такт работы передатчик передает несколько битов исходных данных, например 2 бита при наличии четырех различимых состояний сигнала (рис. 7.13, *б*).

Хотя в формуле Найквиста наличие шума в явном виде не учитывается, косвенно его влияние отражается в выборе количества состояний информационного сигнала. Для повышения пропускной способности линии связи следовало бы увеличивать количество состояний, но на практике этому препятствует шум на линии. Например, пропускную способность линии, сигнал которой показан на рис. 7.13, *б*, можно увеличить еще в два раза, применив для кодирования данных не 4, а 16 уровней. Однако если амплитуда шума время от времени будет превышать разницу между соседними уровнями, то приемник не сможет устойчиво

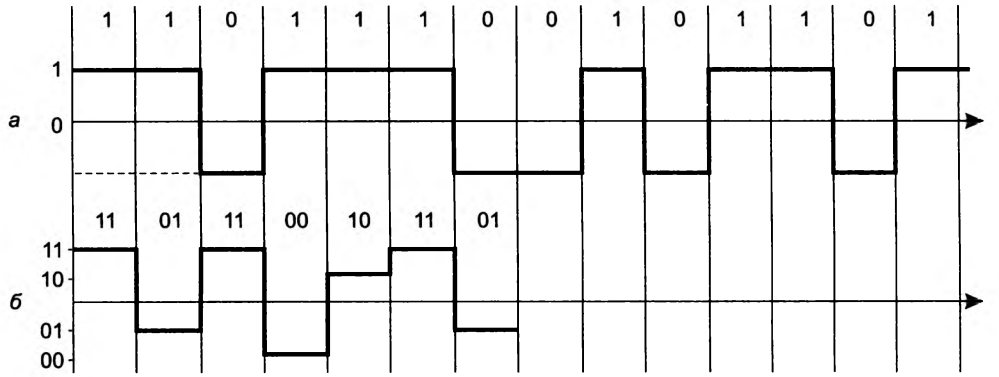


Рис. 7.13. Повышение скорости передачи за счет дополнительных состояний сигнала

распознавать передаваемые данные. Поэтому количество возможных состояний сигнала фактически ограничивается соотношением мощности сигнала и шума, а формула Найквиста определяет предельную скорость передачи данных в том случае, когда количество состояний уже выбрано с учетом возможностей устойчивого распознавания приемником.

Типы кабелей

Сегодня как для внутренней проводки (кабели зданий), так и для внешней чаще всего применяются три класса проводных линий связи:

- витая пара;
- коаксиальные кабели;
- волоконно-оптические кабели.

Экранированная и неэкранированная витая пара

Витой парой называется скрученная пара проводов. Этот вид среды передачи данных очень популярен и составляет основу большого количества как внутренних, так и внешних кабелей. Кабель может состоять из нескольких скрученных пар (внешние кабели иногда содержат до нескольких десятков таких пар).

Скручивание проводов снижает влияние внешних и взаимных помех на полезные сигналы, передаваемые по кабелю.

Основные особенности конструкции кабелей схематично показаны на рис. 7.14.

Кабели на основе витой пары являются *симметричными*, то есть они состоят из двух одинаковых в конструктивном отношении проводников. Симметричный кабель на основе витой пары может быть как *экранированным*, так и *неэкранированным*.

Нужно отличать *электрическую* изоляцию проводящих жил, которая имеется в любом кабеле, от *электромагнитной* изоляции. Первая состоит из непроводящего диэлектрического слоя — бумаги или полимера, например поливинилхлорида или полистирола. Во втором случае помимо электрической изоляции проводящие жилы помещаются также

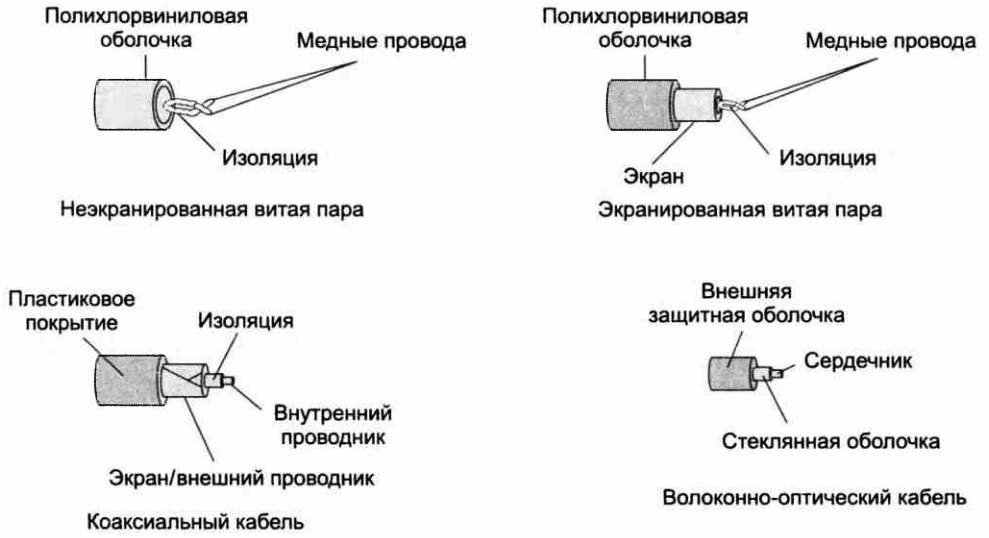


Рис. 7.14. Устройство кабелей

внутрь электромагнитного экрана, в качестве которого чаще всего применяется проводящая медная оплетка.

Экранированная витая пара (Shielded Twisted Pair, **STP**) хорошо защищает передаваемые сигналы от внешних помех, а также меньше излучает электромагнитные колебания вовне, что, в свою очередь, защищает пользователей сетей от вредного для здоровья излучения. Экранироваться может как кабель в целом, так и каждая отдельная пара для уменьшения перекрестных наводок. Наличие заземляемого экрана удорожает кабель и усложняет его прокладку, поэтому экранированную витую пару применяют только в тех случаях, когда это действительно необходимо, например из-за больших внешних помех и повышенных требований к надежности передачи данных.

Неэкранированная витая пара (Unshielded Twisted Pair, **UTP**) обеспечивает защиту от внешних помех только за счет скручивания проводов в пары, что, естественно, не является такой эффективной мерой, как экранирование, но во многих случаях оказывается достаточной для передачи данных с нужным качеством.

Кабели на основе витой пары, используемые для проводки внутри здания, разделяются в международных стандартах на *категории* (от 1 до 7).

Параметры кабелей категорий 1–6 определяются стандартами TIA/EIA-568 (разработанными организацией Telecommunication Industry Association, бывшей долгое время подразделением ныне упраздненной организации Electronic Industries Alliance — от названий этих организаций и происходит аббревиатура TIA/EIA), а также близкими к ним стандартами ISO/IEC 11801 (последние также определяют кабели категории 7, в разработке этой организации на момент написания данной книги находился стандарт на категорию 8).

Все кабели на витой паре независимо от их категории выпускаются в четырехпарном исполнении. Каждая из четырех пар кабеля имеет определенные цвет и шаг скрутки.

Кабели категорий 1–4 сегодня практически не применяются.

Кабели *категории* 5 были специально разработаны для поддержки высокоскоростных протоколов. Их характеристики определяются в диапазоне до 100 МГц. Существует улучшенная версия категории 5e (5 enhanced), которая была разработана специально для более качественной поддержки протокола Gigabit Ethernet, в основном за счет более жестких ограничений на перекрестные наводки.

Появление технологи 10G Ethernet привело к стандартизации более качественных кабелей *категорий* 6, 6*a* и 7. Для кабеля категории 6 характеристики определяются до частоты 250 МГц, категории 6*a* — до 500 МГц, а для кабелей категории 7 — до 600 МГц. Кабели категории 7 обязательно экранируются, причем как каждая пара, так и весь кабель в целом. Кабель категории 6 может быть как экранированным, так и неэкранированным. Максимальная длина сегмента 10G Ethernet на кабеле категории 6 равна 55 м, а на кабелях категорий 6*a* и 7 — 100 м.

Коаксиальный кабель

Коаксиальный кабель состоит из несимметричных пар проводников. Каждая пара представляет собой внутреннюю медную жилу и соосную с ней внешнюю жилу, которая может быть полой медной трубой или оплеткой, отделенной от внутренней жилы диэлектрической изоляцией. Внешняя жила играет двоякую роль — по ней передаются информационные сигналы и она является экраном, защищающим внутреннюю жилу от внешних электромагнитных полей. Существует несколько типов коаксиального кабеля, отличающихся характеристиками и областями применения: для локальных компьютерных сетей, для глобальных телекоммуникационных сетей, для кабельного телевидения и т. п.

Согласно современным стандартам коаксиальный кабель не считается хорошим выбором при построении структурированной кабельной системы зданий. Далее приводятся основные типы и характеристики этих кабелей.

- **«Толстый» коаксиальный кабель** разработан для сетей Ethernet 10Base-5 с волновым сопротивлением 50 Ом и внешним диаметром около 12 мм. Этот кабель имеет достаточно толстый внутренний проводник диаметром 2,17 мм, который обеспечивает хорошие механические и электрические характеристики (затухание на частоте 10 МГц — не хуже 18 дБ/км). Однако этот кабель сложно монтировать — он плохо гнется.
- **«Тонкий» коаксиальный кабель** предназначен для сетей Ethernet 10Base-2. Обладая внешним диаметром около 50 мм и тонким внутренним проводником 0,89 мм, этот кабель не так прочен, как «толстый» коаксиал, зато обладает гораздо большей гибкостью, что удобно при монтаже. «Тонкий» коаксиальный кабель также имеет волновое сопротивление 50 Ом, но его механические и электрические характеристики хуже, чем у «толстого» коаксиального кабеля. Затухание в этом типе кабеля выше, чем в «толстом» коаксиальном кабеле, что приводит к необходимости уменьшать длину кабеля для получения одинакового затухания в сегменте.
- **Телевизионный кабель** с волновым сопротивлением 75 Ом широко применяется в кабельном телевидении. Существуют стандарты локальных сетей, позволяющие использовать такой кабель для передачи данных.
- **Твинаксиальный кабель** по конструкции похож на коаксиальный кабель, но отличается наличием двух внутренних проводников. Такой кабель применяется в новых высокоскоростных стандартах 10G и 100G Ethernet для передачи данных на небольшие рас-

стояния, распараллеливание потоков данных между двумя проводниками упрощает достижение высокой суммарной скорости.

Волоконно-оптический кабель

Волоконно-оптический кабель состоит из тонких (5–60 микрон) гибких стеклянных волокон (волоконных световодов), по которым распространяются световые сигналы. Это наиболее качественный тип кабеля — он обеспечивает передачу данных с очень высокой скоростью (до 100 Гбит/с и выше) на большие расстояния (80–100 км без промежуточного усиления), к тому же он лучше других типов передающей среды обеспечивает защиту данных от внешних помех (в силу особенностей распространения света такие сигналы легко экранировать).

Каждый световод состоит из центрального проводника света (сердцевины) — стеклянного волокна, и стеклянной оболочки, обладающей меньшим показателем преломления, чем сердцевина. Распространяясь по сердцевине, лучи света не выходят за ее пределы, отражаясь от покрывающего слоя оболочки, так как она имеет более низкий коэффициент преломления. В зависимости от распределения показателя преломления и величины диаметра сердечника различают:

- многомодовое волокно со ступенчатым изменением показателя преломления (рис. 7.15, *а*);
- многомодовое волокно с плавным изменением показателя преломления (рис. 7.15, *б*);
- одномодовое волокно (рис. 7.15, *в*).

Понятие «мода» описывает режим распространения световых лучей в сердцевине кабеля.

В **одномодовом кабеле** (Single Mode Fiber, SMF) используется центральный проводник очень малого диаметра, соизмеримого с длиной волны света, — от 5 до 10 мкм. При этом практически все лучи света распространяются вдоль оптической оси световода, не отражаясь от внешнего проводника. Изготовление сверхтонких качественных волокон для одномодового кабеля представляет собой сложный технологический процесс, что делает одномодовый кабель достаточно дорогим. Кроме того, в волокно такого маленького диаметра достаточно сложно направить пучок света, не потеряв при этом значительную часть его энергии. Одномодовый кабель обладает очень низким затуханием — примерно –0,2 дБ/км для окна прозрачности волны размером в 1550 нм.

В **многомодовых кабелях** (Multi Mode Fiber, MMF) используются более широкие внутренние сердечники, которые легче изготовить технологически. В многомодовых кабелях во внутреннем проводнике одновременно существует несколько световых лучей, отражающихся от внешнего проводника под разными углами. Угол отражения луча называется **модой** луча. В многомодовых кабелях с плавным изменением коэффициента преломления режим отражения лучей имеет сложный характер. Возникающая при этом интерференция ухудшает качество передаваемого сигнала, что приводит к искажениям передаваемых импульсов. По этой причине технические характеристики многомодовых кабелей хуже, чем одномодовых.

Учитывая это, многомодовые кабели применяют в основном для передачи данных на скоростях не более 10 Гбит/с на небольшие расстояния (до 300–2000 м), а одномодовые — для передачи данных со сверхвысокими скоростями до сотен гигабитов в секунду (а при использовании технологии DWDM — до нескольких терабитов в секунду) на расстояния до нескольких десятков и даже сотен километров (дальняя связь).

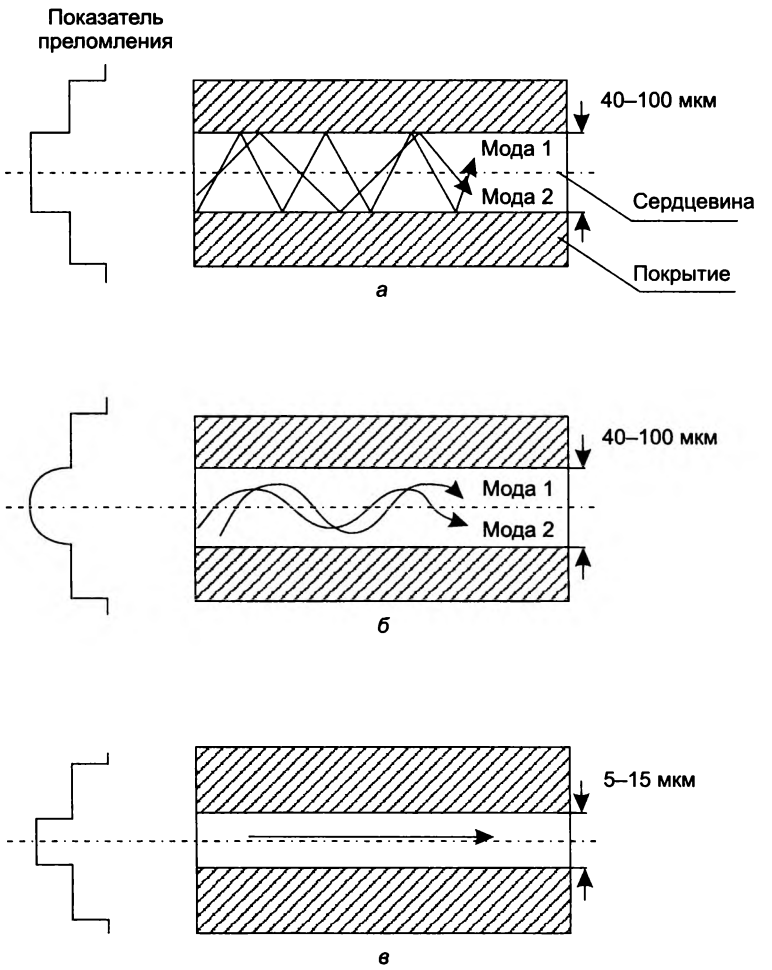


Рис. 7.15. Типы оптического кабеля

В качестве источников света в волоконно-оптических кабелях применяются:

- светодиоды, или светоизлучающие диоды (Light Emitted Diode, LED);
- полупроводниковые лазеры, или лазерные диоды.

Для одномодовых кабелей применяются только лазерные диоды, так как при таком малом диаметре оптического волокна световой поток, создаваемый светодиодом, невозможно без больших потерь направить в волокно — он имеет чересчур широкую диаграмму направленности излучения, в то время как лазерный диод — узкую. Более дешевые светодиодные излучатели используются только для многомодовых кабелей.

Стоимость волоконно-оптических кабелей ненамного превышает стоимость кабелей на витой паре, но проведение монтажных работ с оптоволокном обходится намного дороже из-за трудоемкости операций и высокой стоимости применяемого монтажного оборудования.

Несмотря на отличные характеристики передачи световых сигналов, волоконно-оптические кабели не являются, естественно, идеальными средами и вносят искажения в передаваемый сигнал.

Искажения сигнала в волоконно-оптических кабелях имеют как линейный, так и нелинейный характер (линейность в данном случае определяется по отношению к интенсивности светового сигнала).

К линейным искажениям относятся:

- **Затухание оптического сигнала**. Мощность сигнала уменьшается из-за поглощения света материалом волокна и примесями, рассеивания света из-за неоднородности плотности волокна, а также из-за кабельных искажений, обусловленных деформацией волокон при прокладке кабеля. Затухание измеряется в дБ/км, имеет типичные значения от −0,2 до −0,3 (диапазон 1550 нм), от −0,4 до −1 (диапазон 1310 нм) и от −2 до −3 (диапазон 880 нм).
- **Хроматическая дисперсия** (Chromatic Dispersion, **CD**). Сигнал искажается из-за того, что волны различной длины распространяются вдоль волокна с различной скоростью. Так как прямоугольный импульс имеет спектр ненулевой ширины, из-за хроматической дисперсии составляющие его волны приходят на выход волокна с различной задержкой и фронты импульса оказываются «размытыми». Две составляющие вносят свой вклад в хроматическую дисперсию: **материальная**, отражающая зависимость коэффициента преломления материала сердечника от длины волны, и **волноводная**, вызванная различным поведением волн различной длины на границе между сердечником и оболочкой, то есть там, где изменяется коэффициент преломления. Хроматическая дисперсия оценивается отношением разницы времени распространения двух волн (в пикосекундах) в волокне определенной длины, обычно 1 км, к разнице длин волн (в нанометрах), то есть в пк/нм × км. Материальная составляющая хроматической дисперсии является положительной для волн в окне прозрачности 880 нм (то есть в этом диапазоне длинные волны распространяются быстрее) и отрицательной для волн в окне прозрачности 1550 нм. В окне 1310 нм волны имеют близкую к нулю дисперсию, при этом нуль достигается непосредственно в окрестности волны 1310 нм (такая волна называется длиной волны нулевой дисперсии λ_0 данного кабеля). Типичные значения хроматической дисперсии для окна 1310 нм не превышают 3–5 пс/нм × км, а для окна 1550 нм — 20–25 пс/нм × км.
- **Поляризационная модовая дисперсия** (Polarization Mode Dispersion, **PMD**). Световая мода имеет две взаимно перпендикулярные поляризационные составляющие. В волноводе с идеальным поперечным сечением, то есть представляющим собой окружность, эти составляющие распространяются с одинаковой скоростью. Так как реальные волноводы всегда имеют некоторую овальность, то скорости составляющих отличаются, что приводит к поляризационной дисперсии. Этот вид дисперсии растет пропорционально квадратному корню длины кабеля, поэтому измеряется в пикосекундах, отнесенных к квадратному корню длины кабеля, то есть в пс/$\sqrt{}$км. Типичные значения **PMD** лежат в диапазоне 0,1–0,5 пс/$\sqrt{}$км. Поляризационная дисперсия вносит меньший вклад в искажения оптических импульсов, чем хроматическая, но ее вклад возрастает при увеличении частоты модуляции сигнала.

Нелинейные искажения имеют различную природу, они обусловлены зависимостью коэффициента преломления среды от интенсивности света (эффект Керра), а также эффек-

тами рассеяния света в оптическом волокне (рассеяние Рамана, рассеяние Бриллюэна). Нелинейная зависимость таких эффектов от интенсивности светового потока затрудняет их компенсацию, на практике это проявляется в ограничении длины секций волоконно-оптических сетей без преобразования оптического сигнала в электрический и обратно (такая операция называется регенерацией оптического сигнала).

Стандарты волоконно-оптических кабелей разрабатываются в ITU-T. Рекомендация G.652 описывает характеристики так называемого стандартного одномодового волоконно-оптического кабеля; его «стандартность» заключается в том, что волна нулевой дисперсии должна иметь длину в диапазоне от 1300 до 1324 нм.

Одномодовый волоконно-оптический кабель со смещенной нулевой дисперсией определен **рекомендацией G.653**. Такой кабель предназначен для передачи оптического сигнала на какой-либо одной волне из диапазона прозрачности 1550 нм; его волна нулевой дисперсии сдвинута в эту область за счет особой формы изменения показателя преломления в сечении волокна.

Одномодовый волоконно-оптический кабель со смещенной ненулевой дисперсией по стандарту G.655 предназначен для передачи сигнала систем с уплотненным волновым мультиплексированием (Dense Wave Division Multiplexing, DWDM), в которых информация передается несколькими близкими по длине волнами из диапазона 1550 нм (эта технология рассматривается в главе 10). Смещение, как и в случае кабелей стандарта G.653, достигается за счет профиля показателя преломления. Кабели по стандарту G.655 имеют в этом диапазоне хоть и ненулевую, но небольшую и, главное, приблизительно одинаковую величину хроматической дисперсии, так что волны разной длины задерживаются относительно друг друга не так значительно, как при использовании кабелей G.652 или G.653.

Существует стандарт ITU-T и на многомодовые волоконно-оптические кабели — это рекомендация G.651.1.

Структурированная кабельная система зданий

Структурированная кабельная система (Structured Cabling System, SCS) здания — это набор коммутационных элементов (кабелей, разъемов, коннекторов, кроссовых панелей и шкафов), а также методика их совместного использования, которая позволяет создавать регулярные легко расширяемые структуры связей в вычислительных сетях. Здание как таковое представляет собой достаточно регулярную структуру — оно состоит из этажей, а каждый этаж, в свою очередь, состоит из определенного количества комнат, соединенных коридорами. Структура здания предопределяет структуру его кабельной системы.

Структурированная кабельная система здания — это своего рода «конструктор», с помощью которого проектировщик сети строит нужную ему конфигурацию из стандартных кабелей, соединенных стандартными разъемами и коммутируемых на стандартных кроссовых панелях. При необходимости конфигурацию связей можно легко изменить — добавить компьютер, сегмент, коммутатор, изъять ненужное оборудование, поменять соединение между компьютером и концентратором.

Наиболее детально на сегодня разработаны стандарты кабельных систем зданий, при этом иерархический подход к процессу создания такой кабельной системы позволяет назвать ее структурированной. На основе SCS здания работает одна или несколько локальных сетей организаций или подразделений одной организации, размещенной в этом здании.

SCS планируется и строится иерархически с главной магистралью и многочисленными ответвлениями от нее (рис. 7.16).

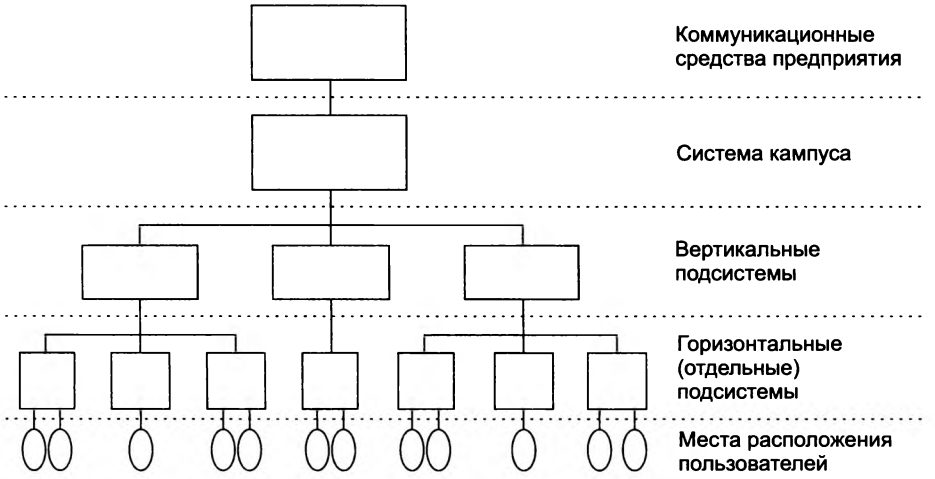


Рис. 7.16. Иерархия структурированной кабельной системы

Типичная иерархия SCS включает (рис. 7.17):

- *горизонтальные подсистемы*, соответствующие этажам здания, — они соединяют кроссовые шкафы этажа с розетками пользователей;
- *вертикальные подсистемы*, соединяющие кроссовые шкафы каждого этажа с центральной аппаратной здания;
- *подсистему кампуса*, объединяющую несколько зданий с главной аппаратной всего кампуса (эта часть кабельной системы обычно называется магистралью).

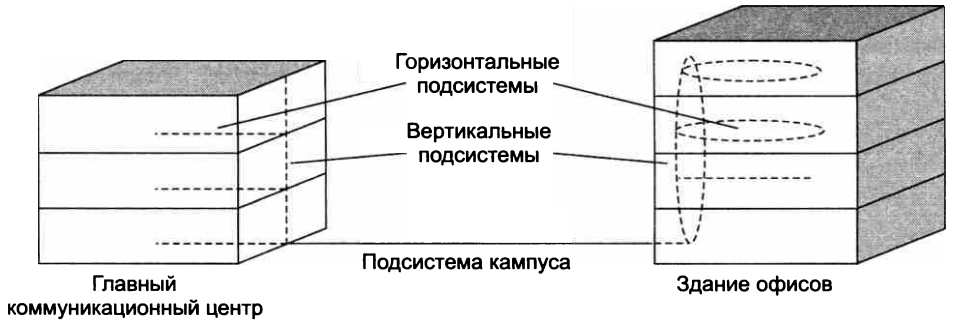


Рис. 7.17. Структура кабельных подсистем

Использование структурированной кабельной системы вместо хаотически проложенных кабелей дает предприятию много преимуществ. SCS при продуманной организации может стать *универсальной средой* передачи компьютерных данных в локальной вычислитель-

ной сети, организации локальной телефонной сети, передачи видеоинформации и даже передачи сигналов от датчиков пожарной безопасности или охранных систем. Подобная универсализация позволяет автоматизировать многие процессы контроля, мониторинга и управления хозяйственными службами и системами жизнеобеспечения предприятия.

Кроме того, применение SCS делает *более экономичным* добавление новых пользователей и изменение их мест размещения. Известно, что стоимость кабельной системы определяется в основном не стоимостью кабеля, а стоимостью работ по его прокладке. Поэтому выгоднее провести однократную работу по прокладке кабеля, возможно, с большим запасом по длине, чем несколько раз выполнять прокладку, наращивая длину кабеля.

Выводы

В зависимости от типа промежуточной аппаратуры все линии связи делятся на аналоговые и цифровые. В аналоговых линиях промежуточная аппаратура предназначена для усиления аналоговых сигналов. В аналоговых линиях используется частотное мультиплексирование.

В цифровых линиях связи передаваемые сигналы имеют конечное число состояний. В таких линиях используется специальная промежуточная аппаратура — регенераторы, которые улучшают форму импульсов и обеспечивают их ресинхронизацию, то есть восстанавливают период их следования. Промежуточная аппаратура мультиплексирования и коммутации первичных сетей работает по принципу временно́го мультиплексирования каналов, когда каждому низкоскоростному каналу выделяется определенная доля времени (тайм-слот, или квант) высокоскоростного канала.

Полоса пропускания определяет диапазон частот, которые передаются линией связи с приемлемым затуханием.

Пропускная способность линии связи зависит от ее внутренних параметров, в частности полосы пропускания, внешних параметров — уровня помех и степени ослабления помех, а также принятого способа кодирования дискретных данных.

Формула Шеннона определяет максимально возможную пропускную способность линии связи при фиксированных значениях полосы пропускания линии и отношения мощности сигнала к шуму.

Формула Найквиста выражает максимально возможную пропускную способность линии связи через полосу пропускания и количество состояний информационного сигнала.

Кабели на основе витой пары делятся на неэкранированные (UTP) и экранированные (STP). Кабели UTP проще в изготовлении и монтаже, зато кабели STP обеспечивают более высокий уровень защищенности.

Волоконно-оптические кабели обладают отличными электромагнитными и механическими характеристиками, недостаток их состоит в сложности и высокой стоимости монтажных работ.

Структурированная кабельная система представляет собой набор коммуникационных элементов — кабелей, разъемов, коннекторов, кроссовых панелей и шкафов, которые удовлетворяют стандартам и позволяют создавать регулярные легко расширяемые структуры связей.

Контрольные вопросы

1. Назовите два основных типа среды передачи данных.
2. Может ли цифровой канал передавать аналоговые данные?
3. Чем отличаются усилители и регенераторы телекоммуникационных сетей?
4. Какие меры можно предпринять для увеличения информационной скорости звена? Варианты ответов:
   а) уменьшить длину кабеля;
   б) выбрать кабель с меньшим сопротивлением;
   в) выбрать кабель с более широкой полосой пропускания;
   г) применить метод кодирования с более широким спектром.
5. По какой причине в расчетах затухания линий связи предпочитают использовать единицы опорной мощности?

ГЛАВА 8 Кодирование и мультиплексирование данных

Модуляция

Среды передачи данных предоставляют только потенциальную возможность передачи дискретной информации. Для того чтобы передатчик и приемник, соединенные некоторой средой, могли обмениваться информацией, им необходимо договориться о том, какие сигналы будут соответствовать двоичным единицам и нулям дискретной информации. Для представления дискретной информации в среде передачи данных применяются сигналы двух типов: прямоугольные импульсы и синусоидальные волны. В первом случае чаще используют термин «кодирование», а во втором — «модуляция», но также можно встретить употребление этих терминов как синонимов.

Модуляция при передаче аналоговых сигналов

Исторически модуляция начала применяться для *аналоговой информации* и только потом — для дискретной.

Необходимость в модуляции аналоговой информации возникает, когда нужно передать низкочастотный аналоговый сигнал через канал, находящийся в высокочастотной области спектра. Примером такой ситуации является передача голоса по радио или телевидению. Голос имеет спектр шириной примерно в 10 кГц, а радиодиапазоны включают гораздо более высокие частоты, от 30 кГц до 300 мГц. Еще более высокие частоты используются в телевидении. Очевидно, что непосредственно голос через такую среду передать нельзя.

Для решения проблемы амплитуду высокочастотного несущего сигнала изменяют (модулируют) в соответствии с изменением низкочастотного голосового сигнала (рис. 8.1). При этом спектр результирующего сигнала попадает в нужный высокочастотный диапазон. Такой тип модуляции называется **амплитудной модуляцией** (Amplitude Modulation, AM).

В качестве информационного параметра используют не только амплитуду несущего синусоидального сигнала, но и частоту. В этих случаях мы имеем дело с **частотной модуляцией** (Frequency Modulation, FM)[1].

---

[1] Заметим, что при модуляции аналоговой информации фаза как информационный параметр не применяется.

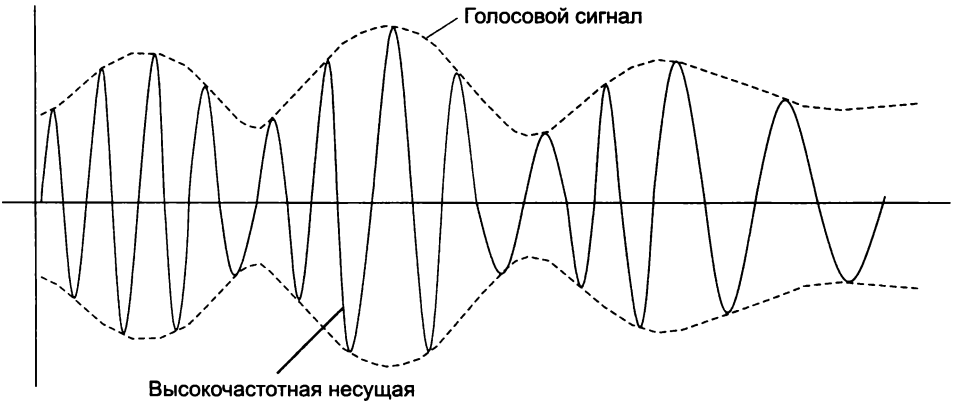


Рис. 8.1. Модуляция голосовым сигналом

Модуляция при передаче дискретных сигналов

При передаче *дискретной информации* посредством модуляции единицы и нули кодируются изменением амплитуды, частоты или фазы несущего синусоидального сигнала. В случае, когда модулированные сигналы передают дискретную информацию, вместо термина «модуляция» иногда используется термин «манипуляция»: амплитудная манипуляция (Amplitude Shift Keying, ASK), частотная манипуляция (Frequency Shift Keying, FSK), фазовая манипуляция (Phase Shift Keying, PSK).

Пожалуй, самый известный пример применения модуляции при передаче дискретной информации — это передача компьютерных данных по телефонным каналам. Типичная амплитудно-частотная характеристика стандартного абонентского канала, называемого также **каналом тональной частоты**, представлена на рис. 8.2. Этот составной канал проходит через коммутаторы телефонной сети и соединяет телефоны абонентов. Канал тональной частоты передает частоты в диапазоне от 300 до 3400 Гц, таким образом, его полоса пропускания равна 3100 Гц. Такая узкая полоса пропускания вполне достаточна для качественной передачи голоса, однако она недостаточно широка для передачи компьютерных данных в виде прямоугольных импульсов. Решение проблемы было найдено благодаря аналоговой модуляции. Устройство, которое выполняет функцию *модуляции* несущей синусоиды на передающей стороне и обратную функцию *демодуляции* на приемной стороне, носит название **модем** (**мод**улятор-**дем**одулятор).

На рис. 8.3 показаны различные типы модуляции, применяемые при передаче дискретной информации. Исходная последовательность битов передаваемой информации приведена на диаграмме, представленной на рис. 8.3, *а*.

При *амплитудной модуляции* для логической единицы выбирается один уровень амплитуды синусоиды несущей частоты, а для логического нуля — другой (рис. 8.3, *б*). Этот способ редко используется в чистом виде на практике из-за низкой помехоустойчивости, но часто применяется в сочетании с другим видом модуляции — фазовой модуляцией.

При *частотной модуляции* значения нуля и единицы исходных данных передаются синусоидами с различной частотой — f_0 и f_1 (рис. 8.3, *в*). Этот способ модуляции не требует сложных

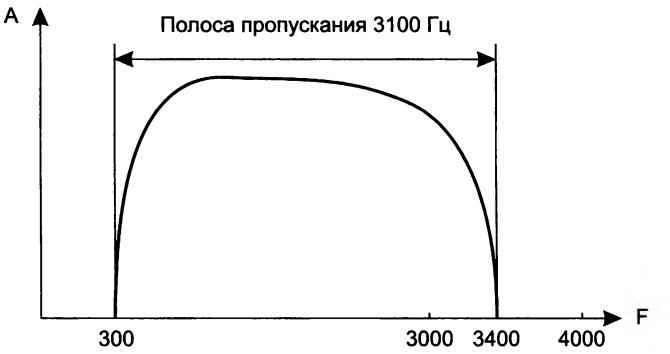


Рис. 8.2. Амплитудно-частотная характеристика канала тональной частоты

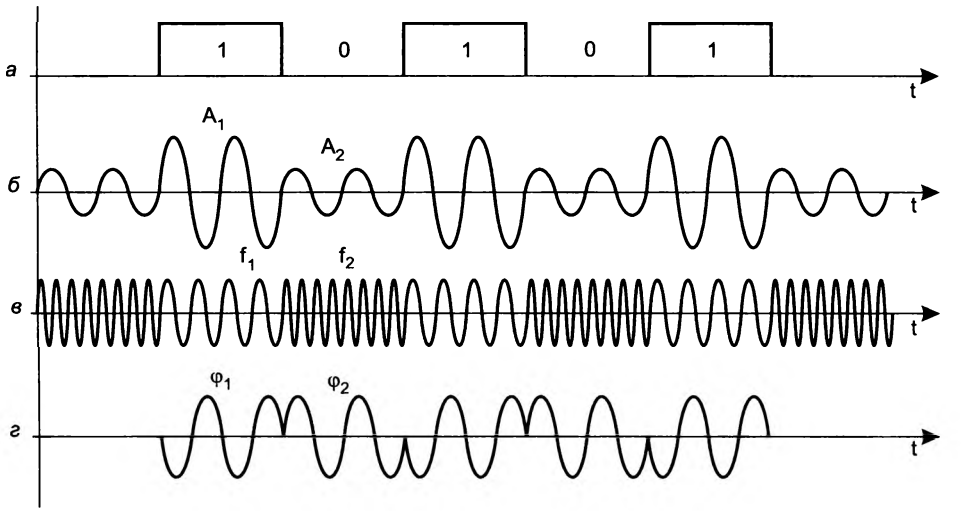


Рис. 8.3. Различные типы модуляции

схем и обычно применяется в низкоскоростных модемах, работающих на скоростях 300 и 1200 бит/с. При использовании только двух частот за один такт передается один бит информации, поэтому такой способ называется **двоичной частотной манипуляцией** (Binary FSK, BFSK). Могут также использоваться четыре различные частоты для кодирования двух битов информации в одном такте, такой способ носит название **четырехуровневой частотной манипуляции** (four-level FSK). Применяется также название **многоуровневая частотная манипуляция** (Multilevel FSK, MFSK).

При **фазовой модуляции** значениям данных 0 и 1 соответствуют сигналы одинаковой частоты, но различной фазы, например 0 и 180° или 0, 90, 180 и 270° (рис. 8.3, *г*). В первом случае такая модуляция носит название **двоичной фазовой манипуляции** (Binary PSK, BPSK), а во втором — **квадратурной фазовой манипуляции** (Quadrature PSK, QPSK).

Комбинированные методы модуляции

Для повышения скорости передачи данных прибегают к комбинированным методам модуляции. Наиболее распространенными являются методы **квадратурной амплитудной модуляции** (Quadrature Amplitude Modulation, QAM). Эти методы основаны на сочетании фазовой и амплитудной модуляции.

На рис. 8.4 показан вариант модуляции, в котором используется 8 различных значений фазы и 4 значения амплитуды. Однако из 32 возможных комбинаций сигнала задействовано только 16, так как разрешенные значения амплитуд у соседних фаз отличаются. Это повышает помехоустойчивость кода, но вдвое снижает скорость передачи данных. Другим решением, повышающим надежность кода за счет введения избыточности, являются так называемые **решетчатые коды**. В этих кодах к каждым четырем битам информации добавляется пятый бит, который даже при наличии ошибок позволяет с большой степенью вероятности определить правильный набор четырех информационных битов.

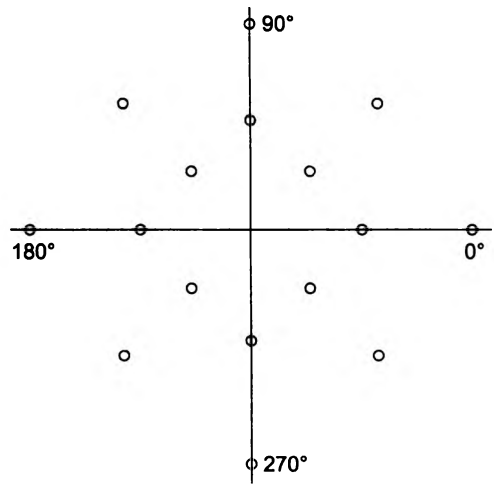


Рис. 8.4. Квадратурная амплитудная модуляция с 16 состояниями сигнала

Спектр модулированного сигнала

Спектр результирующего модулированного сигнала зависит от *типа модуляции* и *скорости модуляции*, то есть частоты изменения модулируемого сигнала (напомним, что она измеряется в бодах).

Рассмотрим сначала *спектр сигнала при потенциальном кодировании*. Эта информация полезна при сравнении характеристик методов модуляции и кодирования.

Пусть логическая единица кодируется положительным потенциалом, а логический ноль – отрицательным потенциалом такой же величины. Для упрощения вычислений предположим, что передается информация, состоящая из бесконечной последовательности чередующихся единиц и нулей, как показано на рис. 8.3, *а*.

Спектр непосредственно получается из формул Фурье для периодической функции. Пусть дискретные данные передаются с тактовой частотой N бод, так как потенциальный код имеет два состояния сигнала, то и битовая скорость передачи данных будет равна N бит/с.

Спектр такого кода состоит из постоянной составляющей нулевой частоты и бесконечного ряда гармоник с частотами f_0, $3f_0$, $5f_0$, $7f_0$, ..., где $f_0 = N/2$. Частота f_0 — первая частота спектра — называется **основной гармоникой**.

Амплитуды этих гармоник убывают достаточно медленно — с коэффициентами 1/3, 1/5, 1/7, ... от амплитуды гармоники f_0 (рис. 8.5, *а*). Заметим, что ширина спектра *прямо пропорциональна тактовой частоте передатчика* N — это свойство полезно знать при выборе тактовой частоты передатчика при заданной полосе пропускания линии.

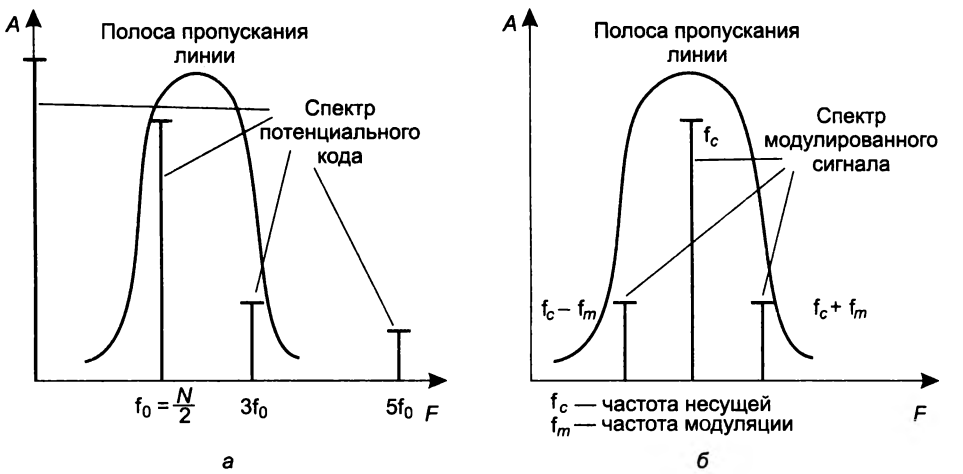


Рис. 8.5. Спектры сигналов при потенциальном кодировании и амплитудной модуляции

В результате спектр потенциального кода требует для качественной передачи широкую полосу пропускания. Кроме того, нужно учесть, что реально спектр сигнала постоянно меняется в зависимости от того, какие данные передаются по линии связи. Например, передача длинной последовательности нулей или единиц сдвигает спектр в сторону низких частот, а в крайнем случае, когда передаваемые данные состоят только из единиц (или только из нулей), спектр состоит из гармоники нулевой частоты. При передаче чередующихся единиц и нулей постоянная составляющая отсутствует. Поэтому спектр результирующего сигнала потенциального кода при передаче произвольных данных занимает полосу от некоторой величины, близкой к нулю, до примерно $7f_0$ (гармониками с частотами выше $7f_0$ можно пренебречь из-за их малого вклада в результирующий сигнал). Для канала тональной частоты верхняя граница при потенциальном кодировании достигается для скорости передачи данных в 971 бит/с, а нижняя неприемлема для любых скоростей, так как полоса пропускания канала начинается с 300 Гц. Поэтому потенциальные коды на каналах тональной частоты никогда не используются.

При амплитудной модуляции спектр состоит из синусоиды несущей частоты f_c, двух боковых гармоник ($f_c + f_m$) и ($f_c - f_m$), а также боковых гармоник ($f_c + 3f_m$) и ($f_c - 3f_m$), где f_m — частота изменения информационного параметра синусоиды, то есть тактовая частота

передатчика. Так как данный метод кодирования использует два уровня амплитуды, то скорость передачи данных также равна f_m (рис. 8.5, *б*). Частота f_m определяет пропускную способность линии при данном способе кодирования. На небольшой частоте модуляции ширина спектра сигнала также оказывается небольшой (равной $2f_m$), если пренебречь гармониками $3f_m$, мощность которых незначительна.

При фазовой и частотной модуляции спектр сигнала получается более сложным, чем при амплитудной модуляции, так как боковых гармоник здесь образуется более двух, но они тоже симметрично расположены относительно основной несущей частоты, а их амплитуды быстро убывают. Ширина спектра сигнала также *пропорциональна тактовой частоте передатчика.*

Дискретизация аналоговых сигналов

В предыдущем разделе мы познакомились с преобразованием дискретной формы представления информации в аналоговую. В этом разделе рассматривается решение обратной задачи — передачи аналоговой информации в дискретной форме. Такая задача решается в системах цифровых телефонии, радио и телевидения.

Как мы уже упоминали в главе 3, начиная с 60-х годов прошлого века голос начал передаваться по телефонным сетям в цифровой форме, то есть в виде последовательности единиц и нулей. Основной причиной такого перехода является невозможность улучшения качества данных, переданных в аналоговой форме, если они существенно исказились при передаче. Сам аналоговый сигнал не дает никаких указаний ни на то, что произошло искажение, ни на то, как его исправить, поскольку форма сигнала может быть любой, в том числе такой, которую зафиксировал приемник. Улучшение же качества линий, особенно территориальных, требует огромных усилий и капиталовложений. Поэтому на смену аналоговой технике записи и передачи звука и изображений пришла цифровая техника. В этой технике используется так называемая **дискретная модуляция** исходных непрерывных во времени аналоговых процессов.

Амплитуда исходной непрерывной функции измеряется с заданным периодом — за счет этого происходит *дискретизация по времени.*

Затем каждый замер представляется в виде двоичного числа определенной разрядности, что означает *дискретизацию по значениям* — непрерывное множество возможных значений амплитуды заменяется дискретным множеством ее значений.

Устройство, которое выполняет подобную функцию, называется **аналого-цифровым преобразователем** (АЦП). Затем замеры передаются по линиям связи в виде последовательности единиц и нулей. При этом применяются те же методы кодирования (с ними мы познакомимся позднее), что и при передаче изначально дискретной информации.

На приемной стороне линии коды преобразуются в исходную последовательность битов, а специальная аппаратура, называемая **цифроаналоговым преобразователем** (ЦАП), производит демодуляцию оцифрованных амплитуд, восстанавливая исходную непрерывную функцию времени.

Дискретная модуляция основана на *теории отображения Найквиста—Котельникова.* В соответствии с этой теорией аналоговая непрерывная функция, переданная в виде последовательности ее дискретных по времени значений, может быть точно восстановлена,

если частота дискретизации была в два или более раз выше, чем частота самой высокой гармоники спектра исходной функции.

Если это условие не соблюдается, то восстановленная функция будет отличаться от исходной.

Преимуществом цифровых методов записи, воспроизведения и передачи аналоговой информации является возможность контроля достоверности считанных с носителя или полученных по линии связи данных. Для этого можно применять те же методы, что и в случае компьютерных данных, — вычисление контрольной суммы, повторная передача искаженных кадров, применение самокорректирующихся кодов.

Для представления голоса в цифровой форме используются различные методы его дискретизации. Наиболее простой метод, в котором применяется частота квантования амплитуды звуковых колебаний в 8000 Гц, уже был кратко рассмотрен в главе 3. Этот метод имеет название **импульсно-кодовой модуляции** (Pulse Code Modulation, PCM).

Обоснование выбранной частоты квантования в методе PCM достаточно простое. Оно объясняется тем, что в аналоговой телефонии для передачи голоса был выбран диапазон от 300 до 3400 Гц, который достаточно качественно передает все основные гармоники собеседников. В соответствии с *теоремой Найквиста—Котельникова* для качественной передачи голоса достаточно выбрать частоту дискретизации, в два раза превышающую самую высокую гармонику непрерывного сигнала, то есть $2 \times 3400 = 6800$ Гц. Выбранная в действительности частота дискретизации 8000 Гц обеспечивает некоторый запас качества. В методе PCM обычно используется 7 или 8 бит кода для представления амплитуды одного замера. Соответственно это дает 127 или 256 градаций звукового сигнала, что оказывается вполне достаточно для качественной передачи голоса.

При использовании метода PCM для передачи одного голосового канала необходима пропускная способность 56 или 64 Кбит/с в зависимости от того, каким количеством битов представляется каждый замер. Если для этих целей применяется 7 бит, то при частоте передачи замеров в 8000 Гц получаем:

$$8000 \times 7 = 56\,000 \text{ бит/с или } 56 \text{ Кбит/с},$$

а для случая 8 бит:

$$8000 \times 8 = 64\,000 \text{ бит/с или } 64 \text{ Кбит/с}.$$

Как вы знаете, стандартным является цифровой канал 64 Кбит/с, который также называется **элементарным каналом цифровых телефонных сетей**; канал 56 Кбит/с применялся на ранних этапах существования цифровой телефонии, когда один бит из байта, отведенного для передачи данных, изымался для передачи номера вызываемого абонента (детали см. в разделе «Сети PDH» главы 10).

Передача непрерывного сигнала в дискретном виде требует от сетей жесткого соблюдения временно́го интервала в 125 мкс (соответствующего частоте дискретизации 8000 Гц) между соседними замерами, то есть требует синхронной передачи данных между узлами сети.

При отсутствии синхронности прибывающих замеров исходный сигнал восстанавливается неверно, что приводит к искажению голоса, изображения или другой мультимедийной информации, передаваемой по цифровым сетям. Так, искажение синхронизации в 10 мс может привести к эффекту «эха», а сдвиги между замерами в 200 мс приводят к невозможности распознавания произносимых слов.

В то же время потеря одного замера при соблюдении синхронности между остальными замерами практически не сказывается на воспроизводимом звуке. Это происходит за счет сглаживающих устройств в цифро-аналоговых преобразователях, работа которых основана на свойстве инерционности любого физического сигнала — амплитуда звуковых колебаний не может мгновенно измениться на значительную величину.

Методы кодирования

Выбор способа кодирования

При выборе способа кодирования нужно одновременно стремиться к достижению нескольких целей:

- минимизировать ширину спектра сигнала, полученного в результате кодирования;
- обеспечивать синхронизацию между передатчиком и приемником;
- обеспечивать устойчивость к шумам;
- обнаруживать и по возможности исправлять битовые ошибки;
- минимизировать мощность передатчика.

Более **узкий спектр сигнала** позволяет на одной и той же линии (с одной и той же полосой пропускания) добиваться более высокой скорости передачи данных.

Как мы видели ранее, спектр сигнала при некотором выбранном методе кодирования пропорционально увеличивается при увеличении тактовой частоты передатчика, например для потенциального кодирования эта зависимость прямо пропорциональна. Поэтому, зафиксировав способ кодирования, мы можем повышать тактовую частоту передатчика и, следовательно, битовую скорость передаваемых дискретных данных до некоторого предела, до тех пор пока спектр сигнала еще помещается в полосу пропускания линии. Более высокой битовой скорости мы при данном методе кодирования достичь не сможем, так как при дальнейшем повышении тактовой частоты передатчика боковые составляющие спектра будут обрезаться линией и сигналы начнут приходить на приемник искаженными, так что приемник не сможет надежно распознавать биты передаваемой информации.

Но если мы применим другой метод кодирования, который при той же тактовой частоте приводит к сигналам более узкого спектра, то, очевидно, сможем повысить тактовую частоту до более высокого предела. И если новый и старый методы кодирования использовали одно и то же число состояний сигнала, то мы добьемся выигрыша в битовой скорости — во столько раз, во сколько при одной и той же частоте спектр нового метода кодирования у́же старого.

Эти соображения иллюстрирует рис. 8.6.

На рисунке показано, как изменяется соотношение полосы пропускания линии связи B и ширины спектра при двух различных методах кодирования (рис. 8.6, *а* и *б*). Предполагается, что в обоих методах используется одно и то же число состояний сигнала, поэтому при одной и той же тактовой частоте битовая скорость передачи данных, обеспечиваемая этими методами кодирования, равна. Пусть при тактовой частоте N она равна C. Как видно

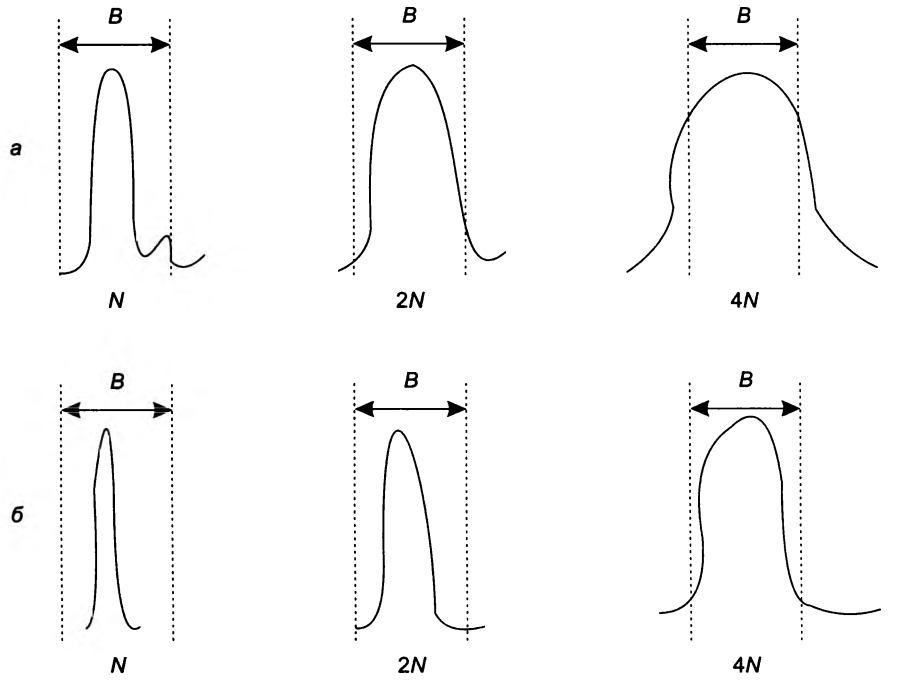


Рис. 8.6. Расширение спектра сигнала в зависимости от увеличения тактовой частоты двух различных методов кодирования (здесь *B* — полоса пропускания линии связи, а *N* — тактовая частота)

из рисунка, ширина спектра метода кодирования первого метода у́же ширины спектра второго при одной и той же тактовой частоте *N*. При этой тактовой частоте спектр сигнала обоих методов кодирования умещается в полосу пропускания линии и оба метода приводят к устойчивой передаче данных со скоростью *C*.

При повышении тактовой частоты в два раза ширина спектра сигнала также увеличилась вдвое, и в обоих случаях она оказалась у́же полосы пропускания линии связи, так что оба метода по-прежнему обеспечивают передачу данных с более высокой скоростью 2*C*. Однако из рисунка видно, что для первого метода такая тактовая частота близка к предельной, так как ширина спектра сигнала практически равна полосе пропускания линии связи. Поэтому повышение тактовой частоты еще в два раза, до 4*N*, для первого метода уже невозможно — его спектр в значительной мере обрезается полосой пропускания линии связи, а значит, сигналы приходят на выход линии сильно искаженными. В то же время второй метод позволяет увеличить тактовую частоту до значения 4*N*, так как его более узкий спектр все еще помещается в полосу пропускания и при таком значении, обеспечивая скорость передачи данных 4*C* бит/с. Очевидно, что второй метод более эффективен для достижения максимальной битовой скорости при фиксированной полосе пропускания линии связи за счет увеличения тактовой частоты передатчика данных.

Синхронизация передатчика и приемника нужна для того, чтобы приемник точно знал, в какой момент времени считывать новую порцию информации с линии связи. При передаче дискретной информации время всегда разбивается на такты одинаковой длительности

и приемник старается считать новый сигнал в середине каждого такта, то есть синхронизировать свои действия с передатчиком.

Проблема синхронизации в сетях решается сложнее, чем при обмене данными между близко расположенными устройствами, например между блоками внутри компьютера. На небольших расстояниях хорошо работает схема, основанная на отдельной *тактирующей линии связи* (рис. 8.7), так что информация снимается с линии данных только в момент прихода тактового импульса. В сетях использование этой схемы вызывает трудности из-за неоднородности характеристик проводников в кабелях. На больших расстояниях неравномерность скорости распространения сигнала может привести к тому, что тактовый импульс придет настолько позже или раньше соответствующего сигнала данных, что бит данных будет пропущен или считан повторно. Другой причиной, по которой в сетях отказываются от использования тактирующих импульсов, является экономия проводников в дорогостоящих кабелях.

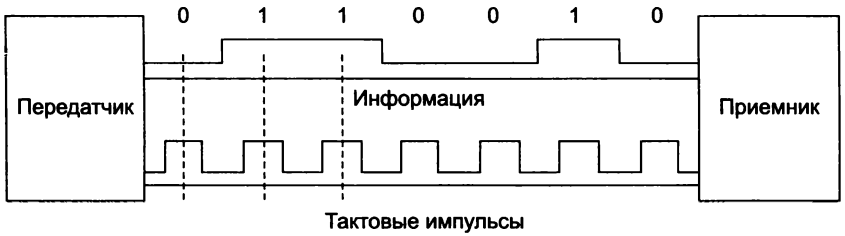


Рис. 8.7. Синхронизация приемника и передатчика на небольших расстояниях

В сетях для решения проблемы синхронизации применяются так называемые **самосинхронизирующиеся коды**, сигналы которых несут для приемника указания о том, в какой момент времени начать распознавание очередного бита (или нескольких битов, если код ориентирован более чем на два состояния сигнала). Любой резкий перепад сигнала — **фронт** — может служить указанием на необходимость синхронизации приемника с передатчиком.

При использовании синусоид в качестве несущего сигнала результирующий код обладает свойством самосинхронизации, так как изменение амплитуды несущей частоты дает возможность приемнику определить момент очередного такта.

Распознавание и коррекцию искаженных данных сложно осуществить средствами физического уровня, поэтому чаще всего эту работу берут на себя вышележащие протоколы: канальный, сетевой, транспортный или прикладной. В то же время распознавание ошибок на физическом уровне экономит время, так как приемник не ждет полного помещения кадра в буфер, а отбраковывает его сразу при распознавании ошибочных битов внутри кадра.

Требования, предъявляемые к методам кодирования, являются взаимно противоречивыми, поэтому каждый из рассматриваемых далее популярных методов кодирования обладает своими достоинствами и недостатками в сравнении с другими.

Потенциальный код NRZ

Рисунок 8.8, *а* иллюстрирует уже упомянутый ранее метод *потенциального кодирования*, называемый также кодированием **без возвращения к нулю** (Non Return to Zero, NRZ).

Последнее название отражает то обстоятельство, что в отличие от других методов кодирования при передаче последовательности единиц сигнал не возвращается к нулю в течение такта.

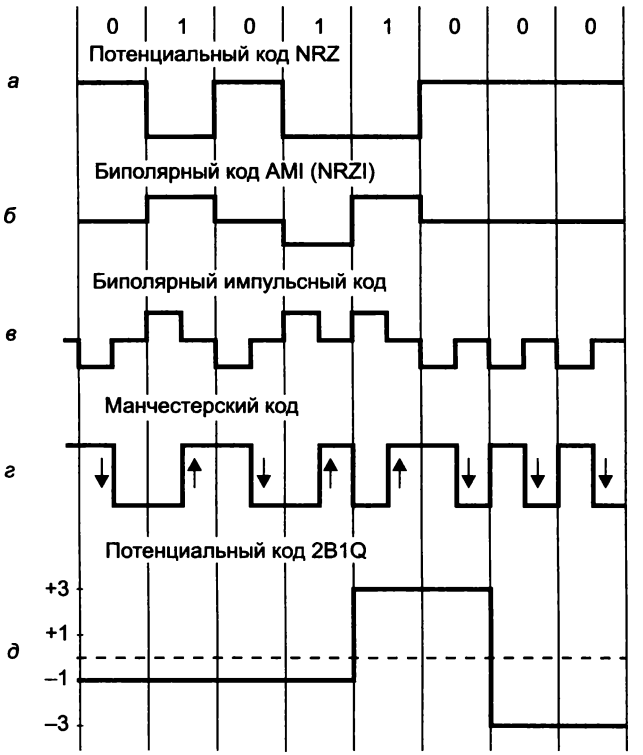


Рис. 8.8. Способы дискретного кодирования данных

Достоинства метода NRZ:

- Простота реализации.
- Хорошая распознаваемость кода (благодаря наличию двух резко отличающихся потенциалов).
- Основная гармоника f_0 имеет достаточно низкую частоту (равную $N/2$ Гц, как было показано в предыдущем разделе), что приводит к относительно узкому спектру.

Недостатки метода NRZ:

- Метод не обладает свойством самосинхронизации. Длинная последовательность единиц или нулей приводит к тому, что сигнал не изменяется в течение многих тактов, так что приемник не имеет возможности синхронизироваться с передатчиком.
- Наличие низкочастотной составляющей, которая приближается к постоянному сигналу при передаче длинных последовательностей единиц или нулей. Из-за этого многие линии связи, не обеспечивающие прямого гальванического соединения между приемником и источником, этот вид кодирования не поддерживают.

Биполярное кодирование AMI

Одной из модификаций метода NRZ является метод **биполярного кодирования с альтернативной инверсией** (Alternate Mark Inversion, AMI). В этом методе применяются три уровня потенциала — отрицательный, нулевой и положительный (см. рис. 8.8, *б*). Для кодирования логического нуля используется нулевой потенциал, а логическая единица кодируется либо положительным потенциалом, либо отрицательным, при этом потенциал каждой новой единицы противоположен потенциалу предыдущей.

При передаче *длинных последовательностей единиц* код AMI частично решает проблемы наличия постоянной составляющей и отсутствия самосинхронизации, присущие коду NRZ. В этих случаях сигнал на линии представляет собой последовательность разнополярных импульсов с тем же спектром, что и у кода NRZ, передающего чередующиеся нули и единицы, то есть без постоянной составляющей и с основной гармоникой $N/2$ Гц (где N — битовая скорость передачи данных). *Длинные же последовательности нулей* для кода AMI столь же опасны, как и для кода NRZ, — сигнал вырождается в постоянный потенциал нулевой амплитуды.

Потенциальный код NRZI

Потенциальный код с инверсией при единице (Non Return to Zero with ones Inverted, NRZI) при передаче нуля сохраняет потенциал, который был установлен на предыдущем такте, а при передаче единицы инвертирует на противоположный.

Код NRZI обладает лучшей самосинхронизацией, чем NRZ, так как при передаче единицы сигнал меняется. Тем не менее при передаче длинных последовательностей нулей сигнал не меняется (например, при передаче последних трех нулей на рис. 8.8, *а*), и значит, у приемника исчезает возможность синхронизации с передатчиком на значительное время, что может приводить к ошибкам распознавания данных.

Биполярный импульсный код

Помимо *потенциальных кодов* в сетях используются *импульсные коды*, в которых данные представлены полным импульсом или же его частью — фронтом. Наиболее простым кодом такого рода является **биполярный импульсный код**, в котором единица представляется импульсом одной полярности, а ноль — другой (см. рис. 8.8, *в*). Каждый импульс длится половину такта. Подобный код обладает отличными самосинхронизирующими свойствами, но постоянная составляющая может присутствовать, например, при передаче длинной последовательности единиц или нулей. Кроме того, спектр у него шире, чем у потенциальных кодов. Так, при передаче всех нулей или единиц частота основной гармоники кода равна N Гц, что в два раза выше основной гармоники кода NRZ и в четыре раза выше основной гармоники кода AMI при передаче чередующихся единиц и нулей. Из-за слишком широкого спектра биполярный импульсный код используется редко.

Манчестерский код

В локальных сетях до недавнего времени самым распространенным был так называемый **манчестерский код** (см. рис. 8.8, *г*). Он применяется в технологии 10 Мбит/с Ethernet.

В манчестерском коде для кодирования единиц и нулей используется перепад потенциала, то есть фронт импульса. При манчестерском кодировании каждый такт делится на две части. Информация кодируется перепадами потенциала, происходящими в середине каждого такта. Единица кодируется перепадом от низкого уровня сигнала к высокому, а ноль — обратным перепадом. В начале каждого такта может происходить служебный перепад сигнала, если нужно представить несколько единиц или нулей подряд. Так как сигнал изменяется по крайней мере один раз за такт передачи одного бита данных, то манчестерский код обладает хорошими самосинхронизирующими свойствами. Полоса пропускания манчестерского кода у́же, чем у биполярного импульсного. Кроме того, у него нет постоянной составляющей, к тому же основная гармоника в худшем случае (при передаче последовательности единиц или нулей) имеет частоту N Гц, а в лучшем (при передаче чередующихся единиц и нулей) — $N/2$ Гц, как и у кодов AMI и NRZ. В среднем ширина полосы манчестерского кода в полтора раза уже, чем у биполярного импульсного кода, а основная гармоника колеблется вблизи значения $3N/4$. Манчестерский код имеет еще одно преимущество перед биполярным импульсным кодом. В последнем для передачи данных используются три уровня сигнала, а в манчестерском — два.

Избыточные коды

Избыточные коды основаны на разбиении исходной последовательности битов на порции, которые часто называют *символами*. Затем каждый исходный символ заменяется новым, с бо́льшим количеством битов, чем исходный.

Например, в логическом коде **4B/5B**, используемом в технологии Fast Ethernet, исходные символы длиной 4 бит заменяются символами длиной 5 бит. Так как результирующие символы содержат избыточные биты, то общее количество битовых комбинаций в них больше, чем в исходных. Так, в коде 4B/5B результирующие символы могут содержать 32 битовые комбинации, в то время как исходные символы — только 16 (табл. 8.1). Поэтому в результирующем коде появляется возможность отобрать 16 таких комбинаций, которые не содержат большого количества нулей, а остальные посчитать **запрещенными кодами** (code violations). Помимо устранения постоянной составляющей и придания коду свойства самосинхронизации избыточные коды позволяют приемнику распознавать искаженные биты. Если приемник принимает запрещенный код, значит, на линии произошло искажение сигнала.

Таблица 8.1. Соответствие исходных и результирующих кодов 4B/5B

Исходный код	Результирующий код	Исходный код	Результирующий код
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

После разбиения получившийся код 4B/5B передается по линии путем преобразования с помощью какого-либо из методов потенциального кодирования, чувствительного только к длинным последовательностям нулей. Таким кодом является, например, NRZI. Символы кода 4B/5B длиной 5 бит гарантируют, что при любом их сочетании на линии не встретятся более трех нулей подряд.

ПРИМЕЧАНИЕ

Буква B в названии кода 4B/5B означает, что элементарный сигнал имеет два состояния (от английского binary — двоичный). Имеются также коды и с тремя состояниями сигнала, например в коде 8B/6T для кодирования 8 бит исходной информации используется код из шести сигналов, каждый из которых имеет три состояния. Избыточность кода 8B/6T выше, чем кода 4B/5B, так как на 256 исходных кодов приходится $3^6 = 729$ результирующих символов.

Использование таблицы перекодировки является очень простой операцией, поэтому этот подход не усложняет сетевые адаптеры и интерфейсные блоки коммутаторов и маршрутизаторов.

Для обеспечения заданной пропускной способности линии передатчик, использующий избыточный код, должен работать с повышенной тактовой частотой. Так, для передачи кодов 4B/5B со скоростью 100 Мбит/с требуется тактовая частота 125 МГц. При этом спектр сигнала на линии расширяется по сравнению со случаем, когда по линии передается не избыточный код. Тем не менее спектр избыточного потенциального кода оказывается у́же спектра манчестерского кода, что оправдывает дополнительный этап логического кодирования, а также работу приемника и передатчика на повышенной тактовой частоте.

Чем ближе к единице соотношение числа исходных символов к общему числу символов, тем незначительнее становится повышение тактовой частоты передатчика. В наиболее скоростных на сегодняшний день версиях 10G Ethernet и 100G Ethernet применяется избыточный код 64B/66B.

Избавиться от длинных последовательностей нулей в коде помогает такой прием, как скремблирование — «перемешивание» битов кода в соответствии с определенным алгоритмом, позволяющим приемнику выполнить обратное преобразование.

(S) *Скремблирование и компрессия данных*

Обнаружение и коррекция ошибок

Надежную передачу информации обеспечивают различные методы. В главе 5 были рассмотрены принципы работы протоколов, которые обеспечивают надежность за счет повторной передачи искаженных или потерянных пакетов. Такие протоколы основаны на том, что приемник в состоянии распознать факт искажения информации в принятом кадре. Еще одним, более эффективным подходом, чем повторная передача пакетов, является использование самокорректирующихся кодов, которые позволяют не только обнаруживать, но и исправлять ошибки в принятом кадре.

Методы обнаружения ошибок

Методы обнаружения ошибок основаны на передаче в составе блока данных избыточной служебной информации, по которой можно судить с некоторой степенью вероятности

о достоверности принятых данных. В сетях с коммутацией пакетов такой единицей информации может быть PDU любого уровня, для определенности будем считать, что мы контролируем кадры.

Избыточную служебную информацию принято называть **контрольной суммой**, или **контрольной последовательностью кадра** (Frame Check Sequence, FCS). Контрольная сумма вычисляется как функция от основной информации, причем *не обязательно путем суммирования*. Принимающая сторона повторно вычисляет контрольную сумму кадра по известному алгоритму и в случае ее совпадения с контрольной суммой, вычисленной передающей стороной, делает вывод о том, что данные были переданы через сеть корректно. Рассмотрим несколько распространенных алгоритмов вычисления контрольной суммы, отличающихся вычислительной сложностью и способностью обнаруживать ошибки в данных.

Контроль по паритету представляет собой наиболее простой метод контроля данных. В то же время это наименее мощный алгоритм контроля, так как с его помощью можно обнаруживать только одиночные ошибки в проверяемых данных. Метод заключается в суммировании по модулю 2 всех битов контролируемой информации. Нетрудно заметить, что для информации, состоящей из нечетного числа единиц, контрольная сумма всегда равна 1, а при четном числе единиц — 0. Например, для данных 100101011 результатом контрольного суммирования будет значение 1. Результат суммирования также представляет собой один дополнительный бит данных, который пересылается вместе с контролируемой информацией. При искажении в процессе пересылки любого одного бита исходных данных (или контрольного разряда) результат суммирования будет отличаться от принятого контрольного разряда, что говорит об ошибке. Однако двойная ошибка, например 110101010, будет неверно принята за корректные данные. Поэтому контроль по паритету применяется к небольшим порциям данных, как правило, к каждому байту, что дает для этого метода коэффициент избыточности 1/8. Метод редко используется в компьютерных сетях из-за значительной избыточности и невысоких диагностических возможностей.

Вертикальный и горизонтальный контроль по паритету представляет собой модификацию описанного метода. Его отличие состоит в том, что исходные данные рассматриваются в виде матрицы, строки которой составляют байты данных. Контрольный разряд подсчитывается отдельно для каждой строки и для каждого столбца матрицы. Этот метод позволяет обнаруживать бóльшую часть двойных ошибок, однако он обладает еще большей избыточностью. На практике этот метод сейчас также почти не применяется при передаче информации по сети.

Циклический избыточный контроль (Cyclic Redundancy Check, CRC) является в настоящее время наиболее популярным методом контроля в вычислительных сетях (и не только в сетях, например этот метод широко применяется при записи данных на гибкие и жесткие диски). Метод основан на представлении исходных данных в виде одного многоразрядного двоичного числа. Например, кадр стандарта Ethernet, состоящий из 1024 байт, рассматривается как одно число из 8192 бит. Контрольной информацией считается остаток от деления этого числа на известный делитель R. Обычно в качестве делителя выбирается семнадцати- или тридцатитрехразрядное число, чтобы остаток от деления имел длину 16 разрядов (2 байт) или 32 разряда (4 байт). При получении кадра данных снова вычисляется остаток от деления на тот же делитель R, но при этом к данным кадра добавляется содержащаяся в нем контрольная сумма. Если остаток от

деления на R равен нулю, то делается вывод об отсутствии ошибок в полученном кадре, в противном случае кадр считается искаженным.

Этот метод обладает более высокой вычислительной сложностью, но его диагностические возможности гораздо выше, чем у методов контроля по паритету. Метод CRC позволяет обнаруживать все одиночные ошибки, двойные ошибки и ошибки в нечетном числе битов. Кроме того, метод обладает невысокой степенью избыточности. Например, для кадра Ethernet размером 1024 байт контрольная информация длиной 4 байт составляет только 0,4 %.

Методы коррекции ошибок

Техника кодирования, которая позволяет приемнику не только понять, что присланные данные содержат ошибки, но и исправить их, называется **прямой коррекцией ошибок** (Forward Error Correction, FEC). Коды, которые обеспечивают прямую коррекцию ошибок, требуют введения большей избыточности в передаваемые данные, чем коды, только обнаруживающие ошибки.

При применении любого избыточного кода не все комбинации кодов являются разрешенными. Например, контроль по паритету делает разрешенными только половину кодов. Если мы контролируем три информационных бита, то разрешенными 4-битными кодами с дополнением до нечетного количества единиц будут такие коды:

000 1, 001 0, 010 0, 011 1, 100 0, 101 1, 110 1, 111 0

То есть всего 8 кодов из 16 возможных.

Для того чтобы оценить количество дополнительных битов, требуемых для исправления ошибок, нужно знать так называемое расстояние Хемминга между разрешенными комбинациями кода. **Расстоянием Хемминга** называется минимальное число битовых разрядов, в которых отличается любая пара разрешенных кодов. Для схем контроля по паритету расстояние Хемминга равно 2.

Можно доказать, что если мы сконструировали избыточный код с расстоянием Хемминга, равным n, то такой код будет в состоянии распознавать $(n-1)$-кратные ошибки и исправлять $(n-1)/2$-кратные ошибки. Так как коды с контролем по паритету имеют расстояние Хемминга, равное 2, то они могут только обнаруживать однократные ошибки и не могут исправлять ошибки.

Коды Хемминга эффективно обнаруживают и исправляют изолированные ошибки, то есть отдельные искаженные биты, которые разделены большим количеством корректных битов. Однако при появлении длинной последовательности искаженных битов (пульсации ошибок) коды Хемминга не работают.

Пульсации ошибок характерны для *беспроводных каналов*, в которых применяют **сверточные коды**. Поскольку для распознавания наиболее вероятного корректного кода в этом методе применяется решетчатая диаграмма, то такие коды еще называют **решетчатыми**. Эти коды используются не только в беспроводных каналах, но и в модемах.

Методы прямой коррекции ошибок особенно эффективны для технологий физического уровня, которые не поддерживают сложные процедуры повторной передачи данных в случае их искажения. Примерами таких технологий являются SDH и OTN, рассматриваемые в главе 10.

Мультиплексирование и коммутация

Методы кодирования и коррекции ошибок позволяют создать в некоторой среде, например в медных проводах кабеля, линию связи. Однако для эффективного соединения пользователей сети этого недостаточно. Нужно образовать в этой линии отдельные каналы передачи данных, служащие для коммутации информационных потоков пользователей. Для создания пользовательского канала коммутаторы первичных сетей должны поддерживать какую-либо технику мультиплексирования и коммутации. Методы коммутации тесно связаны с выбранным методом мультиплексирования, поэтому здесь они изучаются совместно.

В настоящее время для мультиплексирования каналов используются:

- частотное мультиплексирование (Frequency Division Multiplexing, FDM);
- волновое мультиплексирование (Wave Division Multiplexing, WDM).
- временное мультиплексирование (Time Division Multiplexing, TDM);
- множественный доступ с кодовым разделением (Code Division Multiple Access, CDMA).

Метод TDM используется при коммутации как каналов, так и пакетов. Методы FDM, WDM и CDMA пригодны исключительно для коммутации каналов. Метод CDMA применяется только в технике расширенного спектра и рассматривается в следующей главе, посвященной беспроводной передаче.

Коммутация каналов на основе методов FDM и WDM

Техника **частотного мультиплексирования** (FDM) была разработана для телефонных сетей, но применяется она и для других видов сетей, например первичных сетей (микроволновые каналы) или сетей кабельного телевидения.

Основная идея этого метода состоит в выделении каждому соединению собственного диапазона (полосы) частот в общей полосе пропускания линии связи.

На основе этого диапазона создается **канал**. Данные, передаваемые в канале, модулируются с помощью одного из описанных ранее методов с использованием несущей частоты, принадлежащей диапазону канала. Мультиплексирование выполняется с помощью смесителя частот, а демультплексирование — с помощью узкополосного фильтра, ширина которого равна ширине диапазона канала.

Рассмотрим особенности этого вида мультиплексирования на примере телефонной сети.

На входы FDM-коммутатора поступают исходные сигналы от абонентов телефонной сети. Коммутатор переносит сигнал каждого канала в выделенную каналу полосу частот за счет модуляции новой несущей частоты, принадлежащей этой полосе. Чтобы низкочастотные составляющие сигналов разных каналов не смешивались между собой, полосы делают шириной в 4 кГц, а не в 3,1 кГц, оставляя между ними страховочный промежуток в 900 Гц (рис. 8.9). В линии связи между двумя FDM-коммутаторами одновременно передаются сигналы всех абонентских каналов, но каждый из них занимает *свою* полосу частот. Такой канал называют **уплотненным**.

Выходной FDM-коммутатор выделяет модулированные сигналы каждой несущей частоты и передает их на соответствующий выходной канал, к которому непосредственно подключен абонентский телефон.

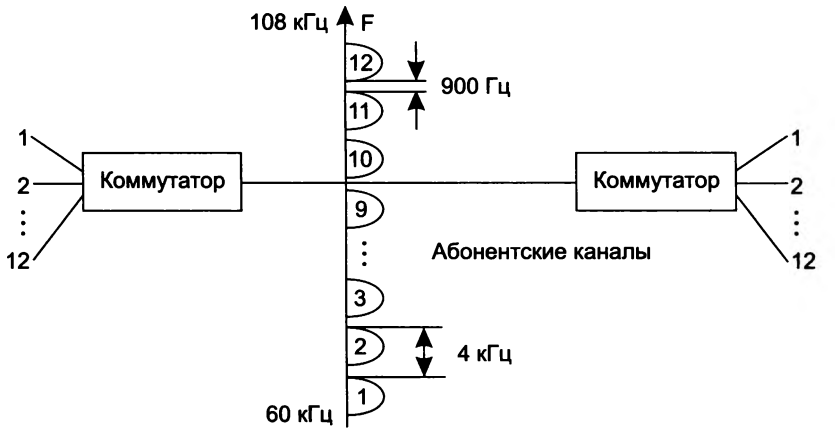


Рис. 8.9. FDM-коммутация

FDM-коммутаторы могут выполнять как динамическую, так и постоянную коммутацию. При *динамической коммутации* один абонент инициирует соединение с другим абонентом, посылая в сеть его номер, и коммутатор выделяет данному абоненту одну из свободных полос своего уплотненного канала *на время сеанса связи* (телефонного разговора). При *постоянной коммутации* администратор сети закрепляет полосу за абонентом на *длительный срок*.

Принцип коммутации на основе разделения частот остается неизменным и в сетях другого вида, меняются только границы полос, выделяемых отдельному абонентскому каналу, а также количество низкоскоростных каналов в высокоскоростном канале.

В методе **волнового мультиплексирования** (WDM) используется тот же принцип частотного разделения каналов, но только в другой области электромагнитного спектра. Информационным сигналом является не электрический ток и не радиоволны, а свет. Для организации WDM-каналов в волоконно-оптическом кабеле задействуют волны инфракрасного диапазона длиной от 850 до 1565 нм, что соответствует частотам от 196 до 350 ТГц.

В магистральном канале обычно мультиплексируется несколько спектральных каналов — до 16, 32, 40, 80 или 160, причем начиная с 16 каналов эта техника мультиплексирования называется **уплотненным волновым мультиплексированием** (Dense Wave Division Multiplexing, DWDM). Внутри такого спектрального канала данные могут кодироваться как дискретным способом, так и аналоговым. По сути, WDM и DWDM — это реализации идеи частотного аналогового мультиплексирования, но в другой форме. Отличие сетей WDM/DWDM от сетей FDM заключается в предельных скоростях передачи информации. Если сети FDM обычно обеспечивают на магистральных каналах одновременную передачу до 600 разговоров, что соответствует суммарной скорости в 36 Мбит/с (для сравнения с цифровыми каналами скорость пересчитана из расчета 64 Кбит/с на один разговор), то сети DWDM обеспечивают общую пропускную способность до сотен гигабитов и даже нескольких терабитов в секунду.

Технология DWDM рассматривается в главе 10.

Коммутация каналов на основе метода TDM

FDM-коммутация разрабатывалась в расчете на передачу голосовых аналоговых сигналов. Переход к цифровой форме представления голоса стимулировал разработку новой техники мультиплексирования, ориентированной на дискретный характер передаваемых данных и носящей название **временно́го мультиплексирования** (TDM). Принцип временно́го мультиплексирования заключается в выделении канала каждому соединению на определенный период времени. Применяются два типа временно́го мультиплексирования — асинхронный и синхронный. С **асинхронным режимом TDM** мы уже знакомы — он применяется в сетях с коммутацией пакетов. Каждый пакет занимает канал определенное время, необходимое для его передачи между конечными точками канала. Между различными информационными потоками нет синхронизации, каждый пользователь пытается занять канал тогда, когда у него возникает потребность в передаче информации.

Рассмотрим теперь **синхронный режим TDM**[1]. В этом режиме доступ всех информационных потоков к каналу синхронизируется таким образом, чтобы каждый информационный поток периодически получал канал в свое распоряжение на фиксированный промежуток времени.

Рисунок 8.10 поясняет принцип коммутации каналов на основе техники TDM при передаче голоса.

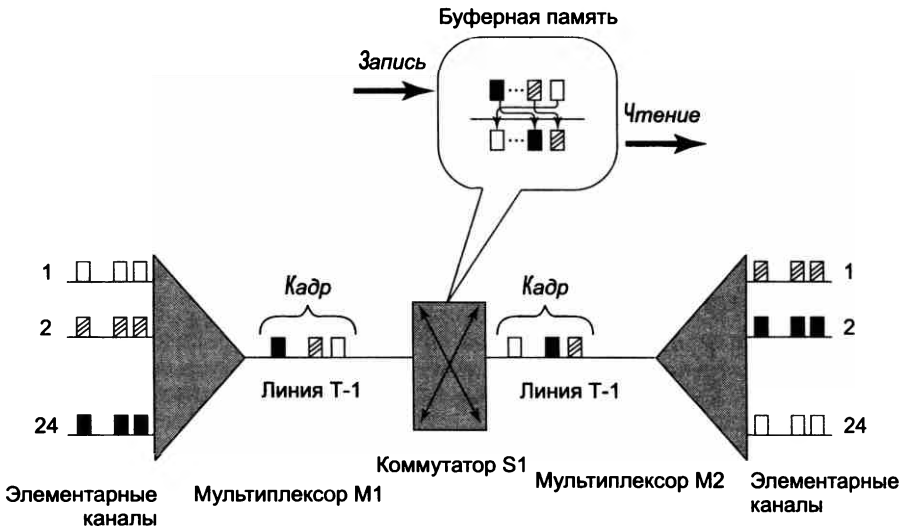


Рис. 8.10. Коммутация на основе разделения канала во времени

Аппаратура сетей TDM — мультиплексоры, коммутаторы, демультиплексоры — работает в режиме разделения времени, поочередно обслуживая в течение цикла своей работы все абонентские каналы. Цикл равен 125 мкс, что соответствует периоду следования замеров

[1] Когда аббревиатура TDM используется без уточнения режима работы, то она всегда обозначает синхронный режим TDM.

голоса в цифровом абонентском канале. Это значит, что мультиплексор или коммутатор успевает вовремя обслужить любой абонентский канал и передать его очередной замер далее по сети. Каждому соединению выделяется один квант времени цикла работы аппаратуры, называемый также **тайм-слотом**. Длительность тайм-слота зависит от числа абонентских каналов, обслуживаемых мультиплексором или коммутатором.

В сети, показанной на рисунке, путем коммутации создано 24 канала, каждый из которых связывает пару абонентов. В частности, абонент, подключенный к входному каналу 1, связан с абонентом, подключенным к выходному каналу 24, абонент входного канала 2 связан с абонентом выходного канала 1, аналогично коммутируются между собой абоненты входного канала 24 и выходного канала 2. Мультиплексор *M*1 принимает информацию от абонентов по входным каналам, каждый из которых передает данные со скоростью 1 байт каждые 125 мкс (64 Кбит/с). В каждом цикле мультиплексор выполняет следующие действия:

1. Прием от каждого канала очередного байта данных.
2. Составление из принятых байтов кадра.
3. Передача кадра на выходной канал с битовой скоростью, равной 24 × 64 Кбит/с, что примерно составляет 1,5 Мбит/с.

Порядок следования байта в кадре соответствует номеру входного канала, от которого этот байт получен. Коммутатор *S*1 принимает кадр по скоростному каналу от мультиплексора и записывает каждый байт из него в отдельную ячейку своей буферной памяти, причем в том порядке, в котором байты были упакованы в уплотненный кадр. Для выполнения коммутации байты извлекаются из буферной памяти не в порядке поступления, а в том порядке, который соответствует поддерживаемым в сети соединениям абонентов. В рассматриваемом примере коммутатор *S*1 коммутирует входные каналы 1, 2 и 24 с выходными каналами 24, 2 и 1 соответственно. Для выполнения этой операции первым из буферной памяти должен быть извлечен байт 2, вторым — байт 24, а последним — байт 1. «Перемешивая» нужным образом байты в кадре, коммутатор обеспечивает требуемое соединение абонентов в сети.

Мультиплексор *M*2 решает обратную задачу — он разбирает байты кадра и распределяет их по своим нескольким выходным каналам, при этом он также считает, что порядковый номер байта в кадре соответствует номеру выходного канала.

Работа TDM-оборудования напоминает работу сетей с коммутацией пакетов, так как каждый байт данных можно считать некоторым элементарным пакетом. Однако в отличие от пакета компьютерной сети «пакет» сети TDM не имеет индивидуального адреса. Его адресом является порядковый номер в кадре или номер выделенного тайм-слота в мультиплексоре или коммутаторе. Сети, использующие технику TDM, требуют синхронной работы всего оборудования, что и определило второе название этой техники — **синхронный режим передачи** (Synchronous Transfer Mode, STM).

Нарушение синхронности разрушает требуемую коммутацию абонентов, так как при этом изменяется относительное положение слота, а значит, теряется адресная информация. Поэтому оперативное перераспределение тайм-слотов между различными каналами в TDM-оборудовании невозможно. Даже если в каком-то цикле работы мультиплексора тайм-слот одного из каналов оказывается избыточным, поскольку на входе этого канала в данный момент нет данных для передачи (например, абонент телефонной сети молчит), то он передается пустым.

Сети TDM могут поддерживать режим динамической или постоянной коммутации, а иногда и оба эти режима. Основным режимом цифровых телефонных сетей, работающих на основе технологии TDM, является динамическая коммутация, но они поддерживают также постоянную коммутацию, предоставляя своим абонентам выделенную линию.

Выводы

Для представления дискретной информации применяются сигналы двух типов: прямоугольные импульсы и синусоидальные волны. В первом случае используют термин «кодирование», во втором — «модуляция».

При модуляции дискретной информации единицы и нули кодируются изменением амплитуды, частоты или фазы синусоидального сигнала.

Аналоговая информация может передаваться по линиям связи в цифровой форме. Это повышает качество передачи, так как позволяет применять эффективные методы обнаружения и исправления ошибок, недоступные для систем аналоговой передачи. Для качественной передачи голоса в цифровой форме используется частота оцифровывания в 8 кГц, когда каждое значение амплитуды голоса представляется 8-битным числом. Это определяет скорость голосового канала в 64 Кбит/с.

При выборе способа кодирования нужно одновременно стремиться к достижению нескольких целей: минимизировать возможную ширину спектра результирующего сигнала, обеспечивать синхронизацию между передатчиком и приемником, обеспечивать устойчивость к шумам, обнаруживать и по возможности исправлять битовые ошибки, минимизировать мощность передатчика.

Спектр сигнала является одной из наиболее важных характеристик способа кодирования. Более узкий спектр сигналов позволяет добиваться более высокой скорости передачи данных при фиксированной полосе пропускания среды.

Код должен обладать свойством самосинхронизации, то есть сигналы кода должны содержать признаки, по которым приемник сможет определить, в какой момент времени нужно осуществлять распознавание очередного бита.

При дискретном кодировании двоичная информация представляется различными уровнями постоянного потенциала или полярностью импульса.

Наиболее простым потенциальным кодом является код без возвращения к нулю (NRZ), однако он не является самосинхронизирующимся.

Коды Хэмминга и сверточные коды позволяют не только обнаруживать, но и исправлять многократные ошибки. Эти коды наиболее часто используются для прямой коррекции ошибок (FEC).

Для образования нескольких каналов в линии связи используются различные методы мультиплексирования, включая частотное (FDM), временно́е (TDM) и волновое (WDM) мультиплексирование, а также множественный доступ с кодовым разделением (CDMA). Техника коммутации пакетов сочетается только с методом TDM, а техника коммутации каналов позволяет задействовать любой тип мультиплексирования.

Контрольные вопросы

1. Какие параметры синусоиды изменяются в методе QAM? Варианты ответов:
   а) амплитуда и фаза;
   б) амплитуда и частота;
   в) частота и фаза.
2. Для какой цели в решетчатых кодах добавляется 5-й бит?
3. Сколько битов передает один символ кода, имеющий 12 состояний?
4. Чем логическое кодирование отличается от физического?
5. Каким образом можно повысить скорость передачи данных по кабельной линии связи? Варианты ответов:
   а) сузить спектр сигнала за счет применения другого метода кодирования/модуляции и повысить тактовую частоту сигнала;
   б) применить кабель с более широкой полосой пропускания и повысить тактовую частоту сигнала;
   в) увеличить спектр сигнала за счет применения другого метода кодирования и повысить тактовую частоту сигнала.

ГЛАВА 9 Беспроводная передача данных

Беспроводная среда передачи

Преимущества беспроводных коммуникаций

Возможность передавать информацию без проводов, привязывающих (в буквальном смысле этого слова) абонентов к определенной точке пространства, всегда была очень привлекательной. И как только технические возможности становились достаточными для того, чтобы новый вид беспроводных услуг приобрел две необходимые составляющие успеха — удобство использования и низкую стоимость, — успех ему был гарантирован.

Последнее тому доказательство — **мобильная телефония**. Первый мобильный телефон был изобретен еще в 1910 году Ларсом Магнусом Эрикссоном (Lars Magnus Ericsson). Этот телефон предназначался для автомобиля и был беспроводным только во время движения. Однако в движении им нельзя было пользоваться, для разговора нужно было остановиться, выйти из автомобиля и с помощью длинных жердей присоединить телефон к придорожным телефонным проводам (рис. 9.1). Понятно, что определенные неудобства и ограниченная мобильность воспрепятствовали коммерческому успеху этого вида телефонии.

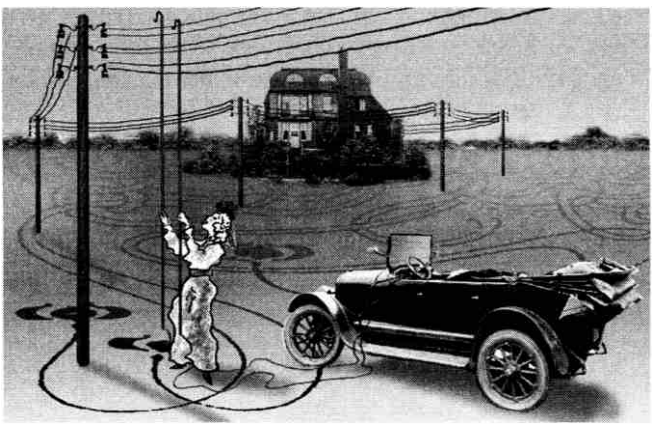


Рис. 9.1. Первый мобильный телефон

Прошло много лет, прежде чем технологии радиодоступа достигли определенной степени зрелости и в конце 70-х обеспечили производство сравнительно компактных и недорогих

радиотелефонов. С этого времени начался бум мобильной телефонии, который продолжается до настоящего времени.

Беспроводная связь не обязательно означает мобильность. Существует так называемая **фиксированная беспроводная связь**, когда взаимодействующие узлы постоянно располагаются в пределах небольшой территории, например в определенном здании. Фиксированная беспроводная связь применяется вместо проводной, когда по какой-то причине невозможно или невыгодно использовать кабельные линии связи. Причины могут быть разными. Например, малонаселенная или труднодоступная местность — болотистые районы и джунгли Бразилии, пустыни, Крайний Север или Антарктида еще не скоро дождутся своих кабельных систем. Другой пример — здания, имеющие историческую ценность, стены которых непозволительно подвергать испытанию прокладкой кабеля. Еще один часто встречающийся случай использования фиксированной беспроводной связи — получение альтернативным оператором связи доступа к абонентам, дома которых уже подключены к точкам присутствия существующего уполномоченного оператора связи проводными линиями доступа. Наконец, организация временной связи, например при проведении конференции в здании, в котором отсутствует проводной канал, имеющий скорость, достаточную для качественного обслуживания многочисленных участников конференции.

Беспроводная связь используется для передачи данных уже достаточно давно. До недавнего времени бóльшая часть применений беспроводной связи в компьютерных сетях была связана с ее фиксированным вариантом. Не всегда архитекторы и пользователи компьютерной сети знают о том, что на каком-то участке пути данные передаются не по проводам, а распространяются в виде электромагнитных колебаний через атмосферу или космическое пространство. Это может происходить в том случае, когда компьютерная сеть арендует линию связи у оператора первичной сети и отдельный канал такой линии является спутниковым или наземным СВЧ-каналом.

Начиная с середины 90-х годов достигла необходимой зрелости и технология **мобильных компьютерных сетей**. С появлением стандарта IEEE 802.11 в 1997 году стало возможным строить мобильные сети Ethernet, обеспечивающие взаимодействие пользователей независимо от того, в какой стране они находятся и оборудование какого производителя они применяют. Пока такие сети еще играют достаточно скромную роль по сравнению с мобильными телефонными сетями, но аналитики предсказывают их быстрый рост в ближайшие годы.

Развитие технологии мобильных телефонных сетей привело к тому, что эти сети стали очень широко использоваться для доступа в Интернет. Третье поколение мобильных телефонных сетей, известное как сети 3G, обеспечивает передачу данных со скоростью 2–10 Мбит/с, что сравнимо по скорости с проводным доступом через телефонные абонентские окончания. В мобильных сетях четвертого поколения 4G предел скорости возрос до 100 Мбит/с (в теории, на практике пока средняя скорость загрузки данных находится в пределах 10–20 Мбит/с).

Беспроводные сети часто связывают с *радиосигналами*, однако это не всегда верно. В беспроводной связи используется широкий диапазон электромагнитного спектра, от радиоволн низкой частоты в несколько килогерц до видимого света, частота которого составляет примерно 8×10^{14} Гц.

Беспроводная линия связи

Беспроводная линия связи строится в соответствии с достаточно простой схемой (рис. 9.2).

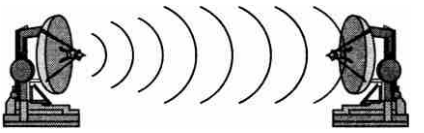


Рис. 9.2. Беспроводная линия связи

Каждый узел оснащается антенной, которая одновременно является передатчиком и приемником электромагнитных волн. Электромагнитные волны распространяются в атмосфере или вакууме со скоростью 3×10^8 м/с во всех направлениях или же в пределах определенного сектора.

Направленность или ненаправленность распространения зависит от типа антенны. На рис. 9.2 показана **параболическая антенна**, которая является *направленной*. Другой тип антенн — **изотропная антенна**, представляющая собой вертикальный проводник длиной в четверть волны излучения. Изотропные антенны являются *ненаправленными*, они широко используются в автомобилях и портативных устройствах. Распространение излучения во всех направлениях можно также обеспечить несколькими направленными антеннами.

Так как при ненаправленном распространении электромагнитные волны заполняют все пространство (в пределах определенного радиуса, определяемого затуханием мощности сигнала), то это пространство может служить *разделяемой средой*. Разделение среды передачи порождает те же проблемы, что и в локальных сетях, однако здесь они усугубляются тем, что пространство в отличие от кабеля является общедоступным, а не принадлежит одной организации.

Кроме того, проводная среда строго определяет направление распространения сигнала в пространстве, а *беспроводная среда является ненаправленной*.

Для передачи дискретной информации с помощью беспроводной линии связи необходимо модулировать электромагнитные колебания передатчика в соответствии с потоком передаваемых битов. Эту функцию осуществляет устройство DCE, располагаемое между антенной и устройством DTE, которым может быть компьютер, коммутатор или маршрутизатор компьютерной сети.

Диапазоны электромагнитного спектра

Характеристики беспроводной линии связи — расстояние между узлами, территория охвата, скорость передачи информации и т. п. — во многом зависят от частоты используемого электромагнитного спектра (частота f и длина волны λ связаны соотношением $c = f \times \lambda$).

На рис. 9.3 показаны диапазоны электромагнитного спектра. Обобщая, можно сказать, что они и соответствующие им беспроводные системы передачи информации делятся на четыре группы.

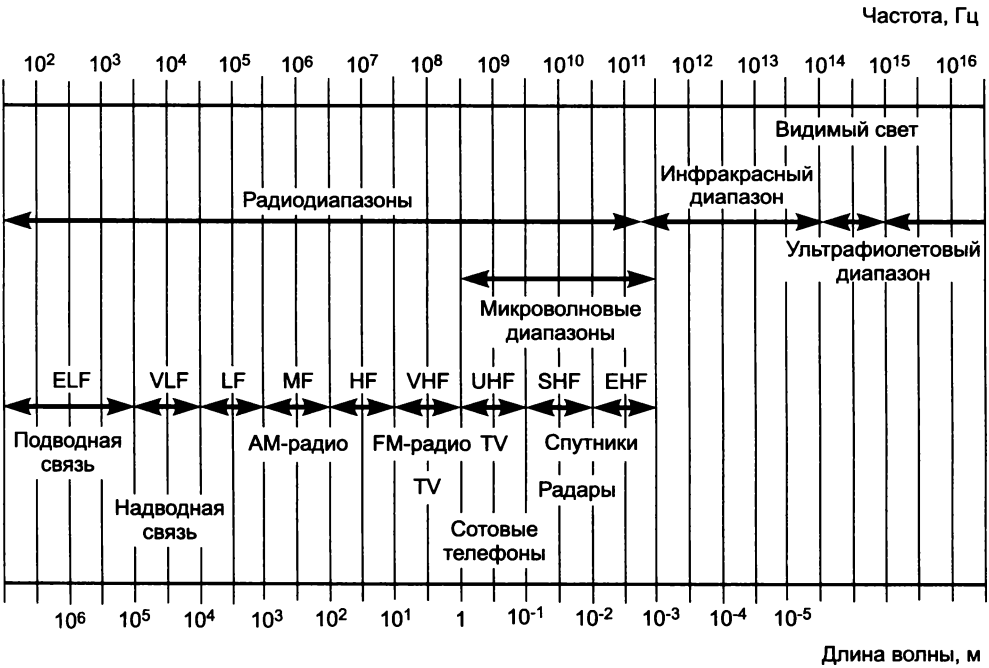


Рис. 9.3. Диапазоны электромагнитного спектра

- Диапазон до 300 ГГц имеет общее стандартное название — **радиодиапазон**. Союз ITU разделил его на несколько поддиапазонов (они показаны на рисунке), начиная от сверхнизких частот (Extremely Low Frequency, ELF) и заканчивая сверхвысокими (Extra High Frequency, EHF). Привычные для нас радиостанции работают в диапазоне от 20 КГц до 300 МГц, и для этих диапазонов существует хотя и не определенное в стандартах, однако часто используемое название **широковещательное радио**. Сюда попадают низкоскоростные системы AM- и FM-диапазонов, предназначенные для передачи данных со скоростями от нескольких десятков до сотен килобит в секунду. Примером могут служить радиомодемы, которые соединяют два сегмента локальной сети на скоростях 2400, 9600 или 19 200 Кбит/с.
- Несколько диапазонов от 300 МГц до 3000 ГГц имеют также нестандартное название микроволновых диапазонов. **Микроволновые системы** представляют наиболее широкий класс систем, объединяющий радиорелейные линии связи, спутниковые каналы, беспроводные локальные сети и системы фиксированного беспроводного доступа, называемые также системами беспроводных абонентских окончаний (Wireless Local Loop, WLL).
- Выше микроволновых диапазонов располагается инфракрасный диапазон. Микроволновые и инфракрасный диапазоны также широко используются для беспроводной передачи информации. Так как инфракрасное излучение не может проникать через стены, то **системы инфракрасных волн** служат для образования небольших сегментов локальных сетей в пределах одного помещения.

- В последние годы видимый свет тоже стал применяться для передачи информации (с помощью лазеров). **Системы видимого света** используются как высокоскоростная альтернатива микроволновым двухточечным каналам для организации доступа на небольших расстояниях.

ПРИМЕЧАНИЕ

Справедливости ради нужно отметить, что свет был, очевидно, первой беспроводной средой передачи информации, так как он использовался в древних цивилизациях (например, в Древней Греции) для эстафетной передачи сигналов между цепочкой наблюдателей, располагавшихся на вершинах холмов.

Распространение электромагнитных волн

Перечислим некоторые общие закономерности распространения электромагнитных волн, связанные с частотой излучения.

Чем выше несущая частота, тем выше возможная скорость передачи информации.

Чем выше частота, тем хуже проникает сигнал через препятствия. Низкочастотные радиоволны АМ-диапазонов легко проникают в дома, позволяя обходиться комнатной антенной. Более высокочастотный сигнал телевидения требует, как правило, внешней антенны. И наконец, инфракрасный и видимый свет не проходят через стены, ограничивая передачу *прямой видимостью* (Line Of Sight, LOS).

Чем выше частота, тем быстрее убывает энергия сигнала с расстоянием от источника. При распространении электромагнитных волн в свободном пространстве (без отражений) затухание мощности сигнала пропорционально произведению квадрата расстояния от источника сигнала на квадрат частоты сигнала.

Низкие частоты (до 2 МГц) распространяются вдоль поверхности земли. Именно поэтому сигналы АМ-радио могут передаваться на расстояния в сотни километров.

Сигналы частот от 2 до 30 МГц отражаются ионосферой Земли, поэтому они могут распространяться даже на более значительные расстояния — в несколько тысяч километров (при достаточной мощности передатчика).

Сигналы в диапазоне выше 30 МГц распространяются только по прямой, то есть являются сигналами прямой видимости. При частоте свыше 4 ГГц их подстерегает неприятность — они начинают поглощаться водой, а это означает, что не только дождь, но и туман может стать причиной резкого ухудшения качества передачи микроволновых систем.

Потребность в скоростной передаче информации является превалирующей, поэтому все современные системы беспроводной передачи информации работают в высокочастотных диапазонах начиная с 800 МГц, несмотря на преимущества, которые сулят низкочастотные диапазоны благодаря распространению сигнала вдоль поверхности Земли или отражения от ионосферы.

Для успешного использования микроволнового диапазона необходимо также учитывать дополнительные проблемы, связанные с поведением сигналов, распространяющихся в режиме прямой видимости и встречающих на своем пути препятствия.

На рис. 9.4 показано, что сигнал, встретившись с препятствием, может распространяться в соответствии с тремя механизмами: отражением, дифракцией и рассеиванием.

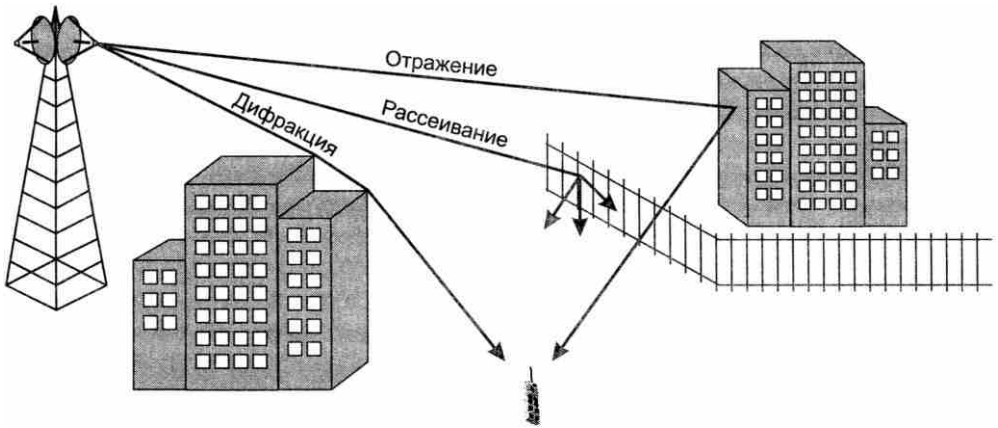


Рис. 9.4. Распространение электромагнитной волны

Когда сигнал встречается с препятствием, которое частично прозрачно для данной длины волны и в то же время размеры которого намного превышают длину волны, то часть энергии сигнала *отражается* от такого препятствия. Волны микроволнового диапазона имеют длину несколько сантиметров, поэтому они частично отражаются от стен домов при передаче сигналов в городе. Если сигнал встречает непроницаемое для него препятствие (например, металлическую пластину) намного большего размера, чем длина волны, то происходит **дифракция** — сигнал как бы огибает препятствие, так что такой сигнал можно получить, даже не находясь в зоне прямой видимости. И наконец, при встрече с препятствием, размеры которого соизмеримы с длиной волны, сигнал *рассеивается*, распространяясь под различными углами.

В результате подобных явлений, которые повсеместно встречаются при беспроводной связи в городе, приемник может получить несколько копий одного и того же сигнала. Такой эффект называется **многолучевым распространением сигнала**. Результат многолучевого распространения сигнала часто оказывается отрицательным, поскольку один из сигналов может прийти в противофазе и подавить основной сигнал.

Так как время распространения сигнала вдоль различных путей является в общем случае различным, то может также наблюдаться **межсимвольная интерференция** — ситуация, когда в результате задержки сигналы, кодирующие соседние биты данных, доходят до приемника одновременно.

Искажения из-за многолучевого распространения приводят к ослаблению сигнала, этот эффект называется **многолучевым замиранием**. В городах многолучевое замирание приводит к тому, что ослабление сигнала становится пропорциональным не квадрату расстояния, а его кубу или даже четвертой степени!

Все эти искажения сигнала складываются с внешними электромагнитными помехами, которых в городе достаточно много. Достаточно сказать, что в диапазоне 2,4 ГГц работают микроволновые печи.

ВНИМАНИЕ

Отказ от проводов и обретение мобильности приводят к высокому уровню помех в беспроводных линиях связи. Если интенсивность битовых ошибок (BER) в проводных линиях связи равна 10^{-9}–10^{-10}, то в беспроводных линиях связи она достигает величины 10^{-3}!

Проблема высокого уровня помех беспроводных каналов решается различными способами. Важную роль играют специальные методы кодирования, распределяющие энергию сигнала в широком диапазоне частот. Кроме того, передатчики сигнала (и приемники, если это возможно) стараются разместить на высоких башнях, чтобы избежать многократных отражений. Еще одним приемом является применение протоколов с установлением соединения и повторными передачами кадров уже на *канальном* уровне стека протоколов. Эти протоколы позволяют быстрее корректировать ошибки, так как работают с меньшими значениями тайм-аутов, чем корректирующие протоколы *транспортного* уровня, такие как TCP.

Лицензирование

Итак, электромагнитные волны могут распространяться во всех направлениях на значительные расстояния и проходить через препятствия, такие как стены домов. Поэтому проблема разделения электромагнитного спектра является весьма острой и требует *централизованного* регулирования. В каждой стране есть специальный государственный орган, который (в соответствии с рекомендациями ITU) выдает **лицензии** операторам связи на использование определенной части спектра, достаточной для передачи информации по определенной технологии. Лицензия выдается на определенную территорию, в пределах которой оператор задействует закрепленный за ним диапазон частот монопольно.

Существуют также три частотных диапазона, 900 МГц, 2,4 ГГц и 5 ГГц, которые рекомендованы ITU как диапазоны для международного использования *без лицензирования*[1]. Эти диапазоны выделены промышленным товарам беспроводной связи общего назначения, например устройствам блокирования дверей автомобилей, научным и медицинским приборам. В соответствии с назначением эти диапазоны получили название **ISM-диапазонов** (Industrial, Scientific, Medical — промышленность, наука, медицина). Диапазон 900 МГц является наиболее «населенным». Это и понятно, низкочастотная техника всегда стоила дешевле. Сегодня активно осваивается диапазон 2,4 ГГц, например в технологиях IEEE 802.11 и Bluetooth. Диапазон 5 ГГц только начал осваиваться, несмотря на то что он обеспечивает более высокие скорости передачи данных.

Обязательным условием использования этих диапазонов на совместной основе является ограничение максимальной мощности передаваемых сигналов уровнем 1 Ватт. Это условие сокращает радиус действия устройств, чтобы их сигналы не стали помехами для других пользователей, которые, возможно, задействуют тот же диапазон частот в других районах города.

Беспроводные системы

Двухточечная связь

Типичная схема проводного двухточечного канала является популярной и для беспроводной связи. По двухточечной схеме могут работать беспроводные каналы различного назначения, использующие различные диапазоны частот.

[1] Диапазоны 900 МГц и 5 ГГц свободны от лицензирования не во всех странах.

В телекоммуникационных первичных сетях эта схема уже долгое время применяется для создания так называемых **радиорелейных линий связи**. Такую линию образуют несколько башен, на которых установлены параболические направленные антенны (рис. 9.5). Каждая линия работает в микроволновом диапазоне на частотах в несколько гигагерц. Направленная антенна концентрирует энергию в узком пучке, что позволяет передавать информацию на значительные расстояние, обычно до 50 км. Высокие башни обеспечивают прямую видимость антенн.

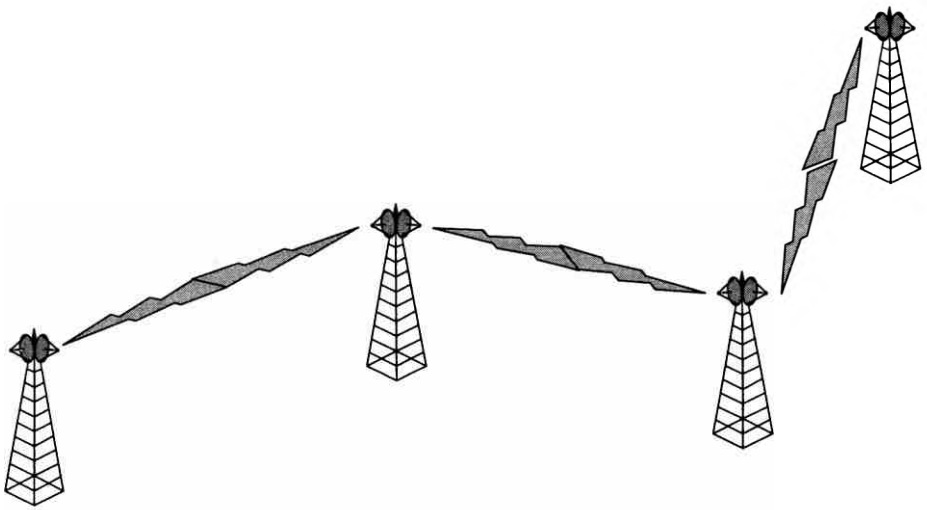


Рис. 9.5. Радиорелейная линия связи

Пропускная способность линии может быть достаточно высокой, обычно она находится в пределах от нескольких до сотен мегабит в секунду. Это могут быть как магистральные линии, так и линии доступа (в последнем случае они имеют чаще всего один канал). Операторы связи часто используют подобные линии, когда прокладка оптического волокна либо невозможна (из-за природных условий), либо экономически невыгодна.

Радиорелейная линия связи может использоваться в городе для соединения двух зданий. Для этого может также использоваться лазер, обеспечивая высокую информационную скорость (до 155 Мбит/с), но только при соответствующем состоянии атмосферы.

Другой пример беспроводной двухточечной линии связи показан на рис. 9.6. Здесь она служит для соединения двух компьютеров. Такая линия образует простейший сегмент локальной сети, поэтому расстояния и мощности сигнала здесь принципиально иные.

Для расстояний в пределах одного помещения может использоваться диапазон инфракрасных волн (рис. 9.6, *а*) или микроволновый диапазон (рис. 9.6, *б*).

Микроволновый вариант работает в пределах нескольких десятков или сотен метров — предельное расстояние предсказать невозможно, так как при распространении микроволнового сигнала в помещении происходят многочисленные отражения, дифракции и рассеивания, к которым добавляются эффекты проникновения волн через стены и межэтажные перекрытия.

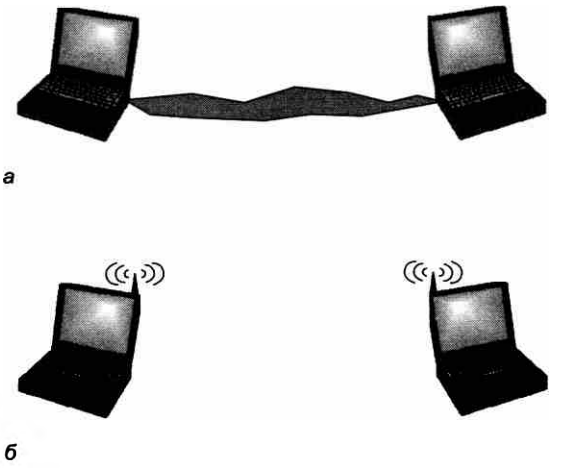


Рис. 9.6. Беспроводная связь двух компьютеров

Связь одного источника и нескольких приемников

Схема беспроводного канала с одним источником и несколькими приемниками характерна для такой организации доступа, при которой многочисленные пользовательские терминалы соединяются с **базовой станцией** (Base Station, BS).

Беспроводные линии связи в схеме с одним источником и несколькими приемниками служат как для фиксированного доступа, так и для мобильного.

На рис. 9.7 показан вариант фиксированного доступа с помощью микроволновых линий связи. Оператор связи использует высокую башню (возможно, телевизионную), чтобы обеспечить прямую видимость с антеннами, установленными на крышах зданий своих клиентов. Фактически такой вариант может представлять собой набор двухточечных линий связи — по количеству зданий, которые необходимо соединить с базовой станцией. Однако это достаточно расточительный вариант, так как для каждого нового клиента нужно устанавливать новую антенну на башне. Поэтому для экономии обычно применяют антенны, захватывающие определенный сектор, например в 45°. Тогда за счет нескольких антенн оператор может обеспечить связь в пределах полного сектора в 360°, конечно, на ограниченном расстоянии (обычно несколько километров).

Пользователи линий доступа могут обмениваться информацией только с базовой станцией, а она, в свою очередь, транзитом обеспечивает взаимодействие между отдельными пользователями.

Базовая станция обычно соединяется проводной связью с проводной частью сети, обеспечивая взаимодействие с пользователями других базовых станций или пользователями проводных сетей. Поэтому базовая станция также называется **точкой доступа** (Access Point, AP). Точка доступа включает не только оборудование DCE, необходимое для образования линии связи, но и чаще всего является коммутатором сети, доступ к которой она обеспечивает, — телефонным коммутатором или коммутатором пакетов.

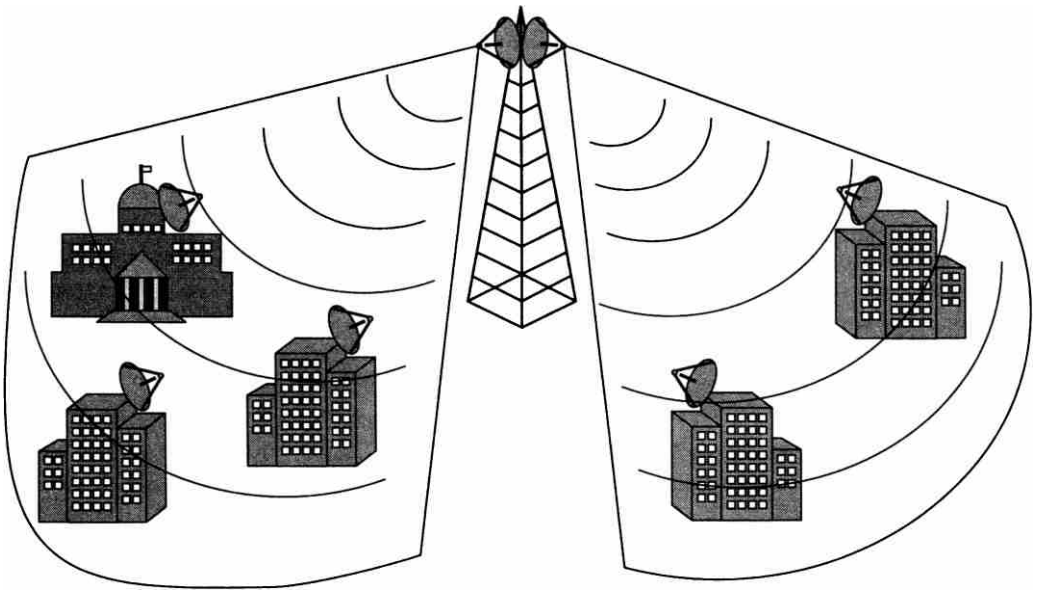


Рис. 9.7. Фиксированный беспроводной доступ

В большинстве схем мобильного доступа используется сегодня принцип **сот**, которые представляют собой небольшие по площади территории, обслуживаемые одной базовой станцией. Идея сот родилась не сразу, первые мобильные телефоны работали по другому принципу, обращаясь к одной базовой станции, покрывающей большую территорию. Идея небольших сот была впервые сформулирована еще в 1945 году, с тех пор прошло довольно много времени, пока заработали первые коммерческие сотовые телефонные сети — пробные участки появились в конце 60-х, а широкое коммерческое применение началось в начале 80-х.

Принцип разбиения всей области охвата сети на небольшие соты дополняется идеей многократного использования частоты. На рис. 9.8 показан вариант организации сот при наличии всего трех частот, при этом ни одна из соседних пар сот не задействует одну и ту же частоту. Многократное использование частот позволяет оператору экономно расходовать выделенный ему частотный диапазон, при этом абоненты и базовые станции соседних сот не испытывают проблем из-за интерференции сигналов. Конечно, базовая станция должна контролировать мощность излучаемого сигнала, чтобы две соты (несмежные), работающие на одной и той же частоте, не создавали друг другу помех.

При гексагональной форме сот количество повторяемых частот может быть больше, чем 3, например 4, 7, 9, 12, 13 и т. д.

Важной проблемой мобильной линии связи является переход терминального устройства из одной соты в другую. Эта процедура, которая называется **эстафетной передачей**, отсутствует при фиксированном доступе и относится к протоколам более высоких уровней, нежели физический.

Поддержка передачи компьютерных данных стала обязательной в мобильных телефонных сетях третьего и четвертого поколений (3G и 4G).

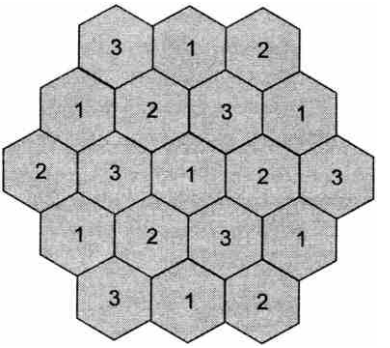


Рис. 9.8. Многократное использование частот в сотовой сети

Связь нескольких источников и нескольких приемников

В случае схемы с несколькими источниками и несколькими приемниками беспроводная линия связи представляет собой общую электромагнитную среду, разделяемую несколькими узлами. Каждый узел может использовать эту среду для взаимодействия с любым другим узлом без обращения к базовой станции. Так как базовая станция отсутствует, то необходим децентрализованный алгоритм доступа к среде.

Чаще всего такой вариант беспроводного канала применяется для соединения компьютеров (рис. 9.9). Для телефонного трафика неопределенность в доле пропускной способности, получаемой при разделении среды, может резко ухудшить качество передачи голоса. Поэтому они строятся по ранее рассмотренной схеме с одним источником (базовой станцией), служащим для распределения полосы пропускания, и несколькими приемниками.

Рис. 9.9. Беспроводная многоточечная линия связи

Собственно, первая локальная сеть, созданная в 70-е годы на Гавайях, в точности соответствовала приведенной на рисунке схеме. Ее отличие от современных беспроводных локальных сетей состояло в низкой скорости передачи данных (9600 бит/с), а также в весьма неэффективном способе доступа, позволяющем использовать только 18 % полосы пропускания.

Децентрализованные многоточечные схемы беспроводного доступа не являются широко распространенными, но в некоторых ситуациях, когда обычная связь с центральной точкой доступа оказывается нерабочей (например, в результате стихийного бедствия, технического отказа сети провайдера или же ее отключения по политической причине), такие схемы оказываются очень востребованными и эффективными. Яркий пример — использование участниками протестов в Гонконге осенью 2014 года приложения для смартфонов FireChat, обеспечивающего децентрализованную маршрутизацию сообщений между телефонами, находящимися в пределах прямой доступности по протоколу BlueTooth или WiFi.

Типы спутниковых систем

Спутниковая связь служит для организации высокоскоростных микроволновых протяженных линий. Так как для таких линий связи нужна прямая видимость, которую из-за кривизны Земли невозможно обеспечить на больших расстояниях, спутник как отражатель сигнала оказывается естественным решением этой проблемы (рис. 9.10).

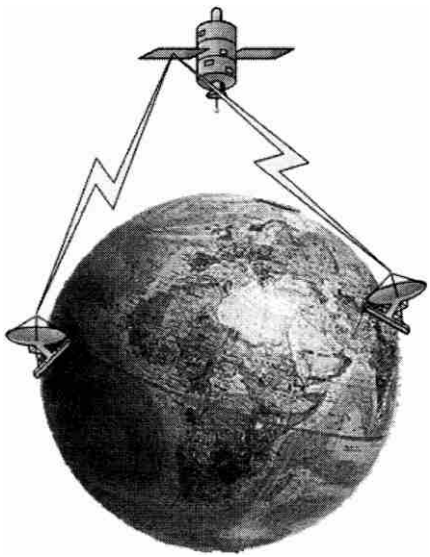


Рис. 9.10. Спутник как отражатель сигнала

Идея использовать искусственный спутник Земли для создания линий связи родилась задолго до запуска в 1957 году первого такого спутника Советским Союзом. Писатель-фантаст Артур Кларк продолжил дело Жюля Верна и Герберта Уэллса, которым удалось описать множество технических изобретений еще до их появления. Кларк в 1945 году описал геостационарный спутник, который висит над одной точкой экватора и обеспечивает связью большую территорию Земли.

Первый спутник, запущенный Советским Союзом в годы холодной войны, обладал очень ограниченными телекоммуникационными возможностями — он только передавал радиосигнал «бип-бип», извещая мир о своем присутствии в космосе. Однако успех России

в космосе подхлестнул усилия Америки, и в 1962 году она запустила первый телекоммуникационный спутник Telstar-1, который поддерживал 600 голосовых каналов.

Со времени запуска первого телекоммуникационного спутника прошло уже более 50 лет, и функции спутника как телекоммуникационного узла, естественно, усложнились. Сегодня спутник может играть роль узла первичной сети, а также телефонного коммутатора и коммутатора/маршрутизатора компьютерной сети. Для этого аппаратура спутников взаимодействует не только с наземными станциями, но и между собой, образуя прямые космические беспроводные линии связи. Принципиально техника передачи микроволновых сигналов в космосе и на Земле не отличается, однако у спутниковых линий связи есть и очевидная специфика — один из узлов такой линии постоянно находится в полете, причем на большом расстоянии от других узлов.

Для спутниковой связи союз ITU выделил несколько *частотных диапазонов*.

Исторически первым использовался диапазон **C**, в котором для каждого из дуплексных потоков Земля—спутник (восходящая частота 4 ГГц) и спутник—Земля (нисходящая частота 6 ГГц) выделяется по 500 МГц — этого достаточно для большого числа каналов. Диапазоны **L** и **S** занимают более низкие частоты и предназначаются для организации мобильных услуг с помощью спутников. Они также часто используются наземными системами.

Спутники отличаются *высотой орбиты над Землей*. Существует три группы орбит (рис. 9.11):

- геостационарная орбита (Geostationary Orbit, GEO) — 35 863 км;
- средневысотная орбита (Medium Earth Orbit, MEO) — 5000–15 000 км;
- маловысотная орбита (Low Earth Orbit, LEO) — 100–1000 км.

Геостационарный спутник «висит» над определенной точкой экватора, в точности следуя скорости вращения Земли. Такое положение выгодно по нескольким обстоятельствам.

Во-первых, четверть поверхности Земли оказывается с такой высоты в зоне прямой видимости, поэтому с помощью геостационарных спутников *просто организовать широковещание в пределах страны или даже континента*.

Во-вторых, геостационарный спутник находится за пределами земной атмосферы и *меньше «изнашивается»*, чем низкоорбитальные и средневысотные спутники. Низкоорбитальные спутники из-за трения о воздух постоянно теряют высоту, и им приходится восстанавливать ее с помощью двигателей.

Путем применения нескольких антенн геостационарные спутники обычно *поддерживают большое количество каналов*.

Наряду с достоинствами у геостационарных спутников есть и недостатки. Наиболее очевидные связаны с *большим удалением спутника от поверхности Земли*. Это приводит к большим задержкам распространения сигнала — от 230 до 280 мс. При использовании спутника для передачи разговора или телевизионного диалога возникают неудобные паузы, мешающие нормальному общению.

Принципиальным недостатком геостационарного спутника с его круговой орбитой является также *плохая связь для районов, близких к Северному и Южному полюсам*. Сигналы в таких районах проходят бóльшие расстояния, чем в районах, расположенных в экваториальных и умеренных широтах, и, естественно, больше ослабляются.

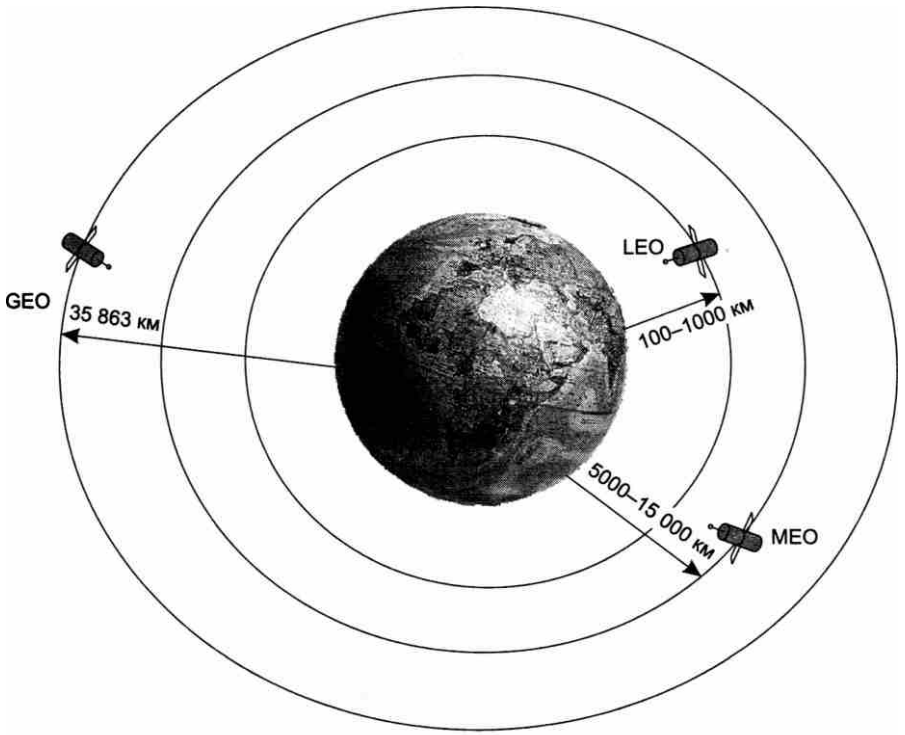


Рис. 9.11. Типы орбит спутников

Среднеорбитальные спутники обеспечивают диаметр покрытия от 10 000 до 15 000 км и задержку распространения сигнала 50 мс. Наиболее известной услугой, предоставляемой спутниками этого класса, является *глобальная система навигации* (Global Positioning System, GPS), известная также под названием NAVigation Satellites providing Time And Range (NAVSTAR). GPS — это всеобщая система определения текущих координат пользователя на поверхности Земли или в околоземном пространстве. GPS состоит из 24 спутников — это то минимальное число спутников, которое необходимо для стопроцентного покрытия территории Земли. Первый тестовый спутник GPS был запущен в 1974 году, первый промышленный спутник — в 1978, а 24-й промышленный — в 1993 году. Спутники GPS летают на орбите высотой около 20 000 км. Помимо спутников в систему GPS входят сеть наземных станций слежения за ними и неограниченное количество пользовательских приемников-вычислителей, среди которых и ставшие очень популярными в последние годы приемники автомобильных систем навигации.

По радиосигналам спутников GPS-приемники пользователей устойчиво и точно определяют координаты; для этого на поверхности Земли приемнику необходимо принять сигналы как минимум от трех спутников. Погрешности не превышают десятков метров. Этого вполне достаточно для решения задач навигации подвижных объектов (самолеты, корабли, космические аппараты, автомобили и т. д.).

В СССР была разработана и реализована система аналогичного назначения под названием ГЛОНАСС (*ГЛО*бальная *НА*вигационная *С*путниковая *С*истема). Первый спутник ГЛО-

НАСС был запущен в октябре 1982 года, а в сентябре 1993 года система была официально введена в эксплуатацию. В 1995 году количество спутников достигло плановой цифры 24 (такое количество необходимо для глобального покрытия Земли, для покрытия территории России достаточно 18 спутников), но затем из-за проблем с финансированием не все выходившие из строя спутники заменялись новыми, поэтому к 2001 году число работающих спутников сократилось до 6. В 2001 была принята новая федеральная программа «Глобальная навигационная система», предусматривающая модернизацию системы спутников к 2008 году и ее полноценную эксплуатацию в 2010 году. С некоторыми задержками эта программа была выполнена, глобальное покрытие было обеспечено в конце 2011 года. Система ГЛОНАСС совместима с GPS, существует навигационное оборудование, которое может принимать сигналы от спутников обеих систем.

Достоинства и недостатки **низкоорбитальных спутников** противоположны соответствующим качествам геостационарных спутников. Главное их достоинство — близость к Земле, а значит, пониженная мощность передатчиков, малые размеры антенн и небольшое время распространения сигнала (около 20–25 мс). Кроме того, их легче запускать. Основной недостаток — малая площадь покрытия, диаметр которой составляет всего около 8000 км. Период оборота такого спутника вокруг Земли составляет 1,5–2 часа, а время видимости спутника наземной станцией — всего 20 минут. Это значит, что постоянная связь с помощью низкоорбитальных спутников может быть обеспечена, только когда на орбите находится достаточно большое их количество. Кроме того, атмосферное трение снижает срок службы таких спутников до 8–10 лет.

Если основным назначением геостационарных спутников является широковещание и дальняя связь, то низкоорбитальные спутники рассматриваются как важное средство поддержания мобильной связи.

В начале 90-х годов достоинства компактных терминальных устройств для низкоорбитальных спутников показались руководителям компании *Motorola* более важными, чем их недостатки. Вместе с несколькими крупными партерами эта компания начала проект *Iridium*, который имел весьма амбициозную цель — создать всемирную спутниковую сеть, обеспечивающую мобильную связь в любой точке земного шара. В конце 80-х еще не существовало такой плотной системы сот мобильной телефонии, как сегодня, так что коммерческий успех казался обеспеченным.

В 1997 группа из 66 спутников была запущена, а в 1998 году началась коммерческая эксплуатация системы *Iridium*. Спутники Iridium действительно покрывают всю поверхность земного шара, вращаясь по 6 орбитам, проходящим через полюсы Земли. На каждой орбите находится по 11 спутников, передатчики которых работают на частоте 1,6 ГГц с полосой пропускания 10 МГц. Эта полоса расходуется 240 каналами по 41 КГц каждый. За счет многократного использования частот система *Iridium* поддерживает 253 440 каналов, организуя системы скользящих по поверхности Земли сот. Для пользователей системы Iridium основным видом услуги является телефонная связь и передача данных со скоростью 2,4 Кбит/с.

К сожалению, коммерческие успехи Iridium оказались очень скромными, и через два года своего существования компания обанкротилась. Расчет на мобильных телефонных абонентов оказался неверным — к моменту начала работы наземная сеть сотовой связи уже покрывала бóльшую часть территории развитых стран. А услуги по передаче данных со скоростью 2,4 Кбит/с не соответствовали потребностям пользователей конца XX века.

Сегодня система Iridium снова работает, теперь уже с новым владельцем и новым именем — *Iridium Communications*. У нее теперь более скромные планы, связанные с созданием местных систем связи в тех частях земного шара, где другая связь практически отсутствует, например на научных станциях Антарктиды. Программное обеспечение спутников модернизируется «на лету», что позволило повысить скорость передачи данных до 10 Кбит/с. В феврале 2008 года компания Iridium Satellite объявила о новой программе под названием Iridium NEXT. В соответствии с этой программой в 2015 году будут запущены новые 66 спутников; все коммуникации со спутниками и между спутниками будут происходить на основе стека протоколов TCP/IP.

В начале 2015 года стадо известно о новой амбициозной инициативе по созданию системы низкоорбитальных спутников, обеспечивающих бесплатный высокоскоростной доступ в Интернет по всему миру. Об этом, в частности, объявил глава компании SpaceX Элон Маск. Его план состоит в запуске 700 недорогих микроспутников весом 110 кг на орбиту высотой 1200 км.

Технология широкополосного сигнала

Техника расширенного спектра разработана специально для беспроводной передачи. Она позволяет повысить помехоустойчивость кода для сигналов малой мощности за счет увеличения спектра передаваемого сигнала, что очень важно в мобильных приложениях. Существует несколько методов расширения спектра.

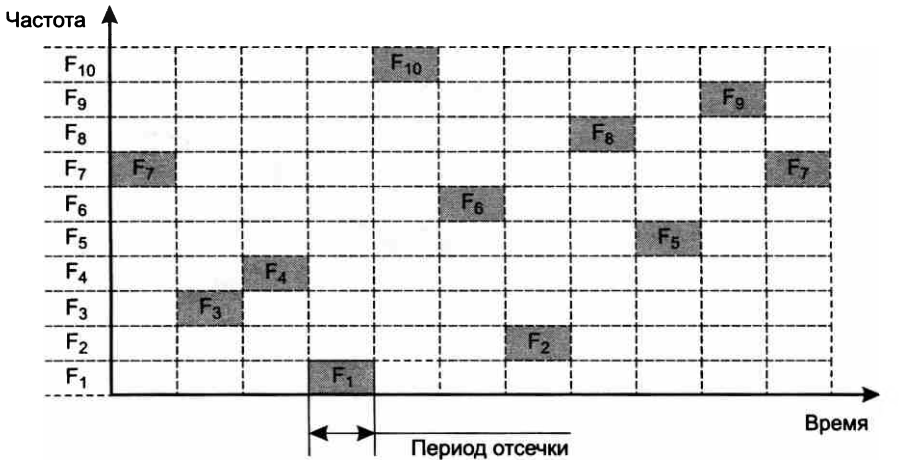
Расширение спектра скачкообразной перестройкой частоты

Идея метода **расширения спектра скачкообразной перестройкой частоты** (Frequency Hopping Spread Spectrum, FHSS) возникла во время Второй мировой войны, когда радио широко использовалось для секретных переговоров и управления военными объектами, например торпедами. Для того чтобы радиообмен нельзя было перехватить или подавить узкополосным шумом, было предложено вести передачу с постоянной сменой несущей в пределах широкого диапазона частот. В результате мощность сигнала распределялась по всему диапазону и прослушивание какой-то определенной частоты давало только небольшой шум. Последовательность несущих частот выбиралась псевдослучайной, известной только передатчику и приемнику. Попытка подавления сигнала в каком-то узком диапазоне также не слишком ухудшала сигнал, так как подавлялась только небольшая часть информации.

Идею этого метода иллюстрирует рис. 9.12.

В течение определенного фиксированного интервала времени передача ведется на неизменной несущей частоте. На каждой несущей частоте для передачи дискретной информации применяются стандартные методы модуляции, такие как FSK или PSK. Чтобы приемник синхронизировался с передатчиком, для обозначения начала каждого периода передачи некоторое время передаются синхробиты. Так что полезная скорость этого метода кодирования оказывается меньше из-за постоянных накладных расходов на синхронизацию.

Несущая частота меняется в соответствии с номерами частотных подканалов, вырабатываемых алгоритмом псевдослучайных чисел. Псевдослучайная последовательность



Последовательность перестройки частот: F_7-F_3-F_4-F_1-F_{10}-F_6-F_2-F_8-F_5-F_9

Рис. 9.12. Расширение спектра скачкообразной перестройкой частоты

зависит от некоторого параметра, который называют **начальным числом**. Если приемнику и передатчику известны алгоритм и значение начального числа, то они меняют частоты в одинаковой последовательности, называемой **последовательностью псевдослучайной перестройки частоты**.

Если частота смены подканалов ниже, чем скорость передачи данных в канале, то такой режим называют **медленным расширением спектра** (рис. 9.13, *а*); в противном случае мы имеем дело с **быстрым расширением спектра** (рис. 9.13, *б*).

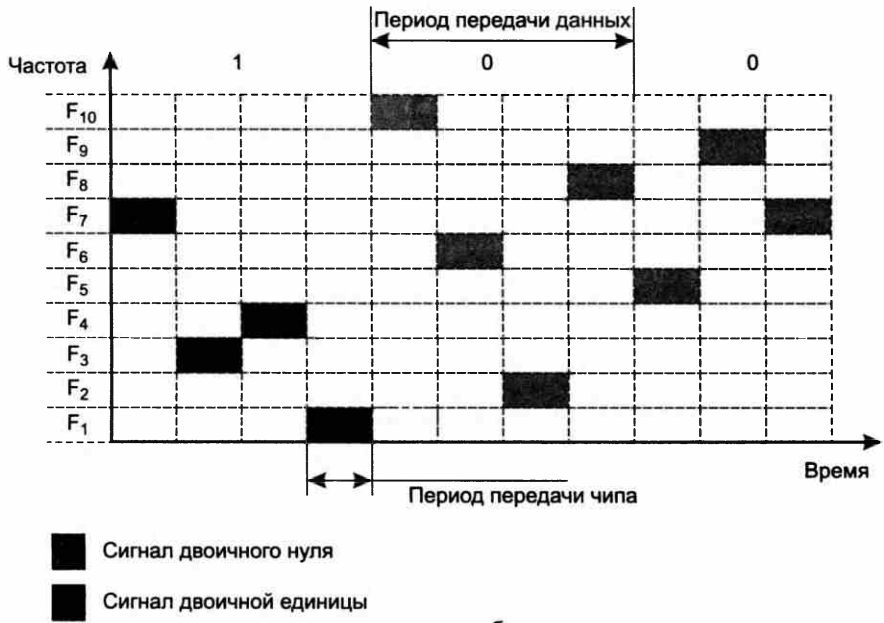
Метод быстрого расширения спектра более устойчив к помехам, поскольку узкополосная помеха, которая подавляет сигнал в определенном подканале, не приводит к потере бита, так как его значение передается несколько раз в различных частотных подканалах. В этом режиме не проявляется эффект межсимвольной интерференции, потому что ко времени прихода задержанного вдоль одного из путей сигнала система успевает перейти на другую частоту.

Метод медленного расширения спектра таким свойством не обладает, но зато он проще в реализации и имеет меньшие накладные расходы.

Методы FHSS применяют в беспроводных технологиях IEEE 802.11 (Wi-Fi) и Bluetooth.

В методах FHSS подход к использованию частотного диапазона не такой, как в других методах кодирования, — вместо экономного расходования узкой полосы делается попытка занять весь доступный диапазон. На первый взгляд это кажется не очень эффективным — ведь в каждый момент времени в диапазоне работает только один канал. Однако последнее утверждение не всегда справедливо, поскольку коды расширенного спектра можно задействовать также и для мультиплексирования *нескольких* каналов в широком диапазоне. В частности, методы FHSS позволяют организовать одновременную работу нескольких каналов путем выбора для каждого канала таких псевдослучайных последовательностей, которые в каждый момент времени дают каждому каналу возможность работать на собственной частоте (конечно, это можно сделать, только если число каналов не превышает числа частотных подканалов).

а



б

Рис. 9.13. Соотношение между скоростью передачи данных и частотой смены подканалов

Прямое последовательное расширение спектра

В методе **прямого последовательного расширения спектра** (Direct Sequence Spread Spectrum, DSSS) частотный диапазон расширяется не за счет постоянных переключений с частоты на частоту, как в методе FHSS, а за счет того, что каждый бит информации заменяется N битами, поэтому тактовая скорость передачи сигналов увеличивается в N раз. А это, в свою очередь, означает, что спектр сигнала также расширяется в N раз. Достаточно соответствующим образом выбрать скорость передачи данных и значение N, чтобы спектр сигнала заполнил весь диапазон.

Цель кодирования методом DSSS та же, что и методом FHSS, — *повышение помехоустойчивости*. Узкополосная помеха будет искажать только определенные частоты спектра сигнала, так что приемник с большой степенью вероятности сможет правильно распознать передаваемую информацию.

Код, которым заменяется двоичная единица исходной информации, называется **расширяющей последовательностью**, а каждый бит такой последовательности — **чипом**. Соответственно скорость передачи результирующего кода называют **чиповой скоростью**. Двоичный нуль кодируется инверсным значением расширяющей последовательности. Приемники должны знать расширяющую последовательность, которую использует передатчик, чтобы понять передаваемую информацию.

Количество битов в расширяющей последовательности определяет **коэффициент расширения** исходного кода. Как и в случае FHSS, для кодирования битов результирующего кода может использоваться любой вид модуляции, например BFSK.

Чем больше коэффициент расширения, тем шире спектр результирующего сигнала и тем больше степень подавления помех. Но при этом растет занимаемый каналом диапазон спектра. Обычно коэффициент расширения имеет значения от 10 до 100.

Примером расширяющей последовательности является *последовательность Баркера* (Barker), которая состоит из 11 бит: 10110111000. Если передатчик использует эту последовательность, то передача трех битов 110 ведет к отправке следующих битов:

10110111000 10110111000 01001000111.

Последовательность Баркера позволяет приемнику быстро синхронизироваться с передатчиком, то есть надежно выявлять начало последовательности. Приемник определяет такое событие, поочередно сравнивая получаемые биты с образцом последовательности. Действительно, если сравнить последовательность Баркера с такой же последовательностью, но сдвинутой на один бит влево или вправо, то мы получим меньше половины совпадений значений битов. Значит, даже при искажении нескольких битов с большой долей вероятности приемник правильно определит начало последовательности, а значит, сможет правильно интерпретировать получаемую информацию.

Метод DSSS в меньшей степени защищен от помех, чем метод быстрого расширения спектра, так как мощная узкополосная помеха влияет на часть спектра, а значит, и на результат распознавания единиц или нулей.

Множественный доступ с кодовым разделением

Как и в случае FHSS, кодирование методом DSSS позволяет мультиплексировать несколько каналов в одном диапазоне. Техника такого мультиплексирования называется

множественным доступом с кодовым разделением (Code Division Multiplexing Access, CDMA). Она широко используется в сотовых сетях.

Хотя техника CDMA может применяться совместно с кодированием методом FHSS, на практике в беспроводной сети она чаще сочетается с методом DSSS.

Каждый узел сети, работающий по методу CDMA, посылает данные в разделяемую среду в те моменты времени, когда это ему нужно, то есть синхронизация между узлами отсутствует. Идея CDMA заключается в том, что каждый узел сети задействует собственное значение расширяющей последовательности. Эти значения выбираются так, чтобы принимающий узел, который знает значение расширяющей последовательности передающего узла, мог выделить данные передающего узла из суммарного сигнала, образующегося в результате одновременной передачи информации несколькими узлами.

Для того чтобы такую операцию демультиплексирования можно было выполнить, значения расширяющей последовательности выбираются определенным образом. Поясним идею CDMA на примере.

Пусть в сети работают четыре узла: *A*, *B*, *C* и *D*. Каждый узел использует следующие значения расширяющей последовательности:

A: 0 1 0 1 0 1 0 1
B: 1 0 1 0 0 0 1 0 1
C: 1 0 0 1 1 0 0 1
D: 1 1 1 1 1 1 1 1

Предположим также, что при передаче единиц и нулей расширяющей последовательности (то есть уже преобразованного исходного кода) используются сигналы, которые являются аддитивными и инверсными. Инверсность означает, что двоичная единица кодируется, например, синусоидой с амплитудой $+A$, а двоичный ноль — синусоидой с амплитудой $-A$. Из условия аддитивности следует, что если фазы этих амплитуд совпадут, то при одновременной передаче единицы и нуля мы получим нулевой уровень сигнала. Для упрощения записи расширяющей последовательности обозначим синусоиду с положительной амплитудой значением +1, а синусоиду с отрицательной амплитудой — значением –1. Для простоты допустим также, что все узлы сети CDMA синхронизированы.

Таким образом, при передаче единицы исходного кода четыре узла передают в среду такие последовательности:

A: –1 +1 –1 +1–1 +1 –1 +1
B: +1 –1 +1 –1 –1 +1 –1 +1
C: +1 –1 –1 +1 +1 –1 –1 +1
D: +1 +1 +1 +1 +1 +1 +1 +1

При передаче нуля исходного кода сигналы расширяющей последовательности инвертируются.

Пусть теперь каждый из четырех узлов независимо от других передает в сеть один бит исходной информации: узел $A \rightarrow 1$, узел $B \rightarrow 0$, узел $C \rightarrow 0$, узел $D \rightarrow 1$.

В среде *S* сети наблюдается такая последовательность сигналов:

A: –1 +1 –1 +1–1 +1 –1 +1
B: –1 +1 –1 +1 +1 –1 +1 –1
C: –1 +1 +1 –1 –1 +1 +1 –1
D: +1 +1 +1 +1 +1 +1 +1 +1

В соответствии со свойством аддитивности получаем:

S: −2 +4 0 +2 0 +2 +2 0

Если, например, некоторый узел E хочет принимать информацию от узла A, то он должен использовать свой демодулятор CDMA, задав ему в качестве параметра значение расширяющей последовательности узла A.

Демодулятор CDMA последовательно складывает все четыре суммарных сигнала S_i, принятые в течение каждого такта работы. При этом сигнал S_i, принятый в такте, на котором код расширения станции A равен +1, учитывается в сумме со своим знаком, а сигнал, принятый в такте, на котором код расширения станции A равен −1, добавляется в сумму с противоположным знаком. Другими словами, демодулятор выполняет операцию скалярного умножения вектора принятых сигналов на вектор значения расширяющей последовательности нужной станции:

$$S \times A = (-2 +4\ 0 +2\ 0 +2 +2\ 0) \times (-1 +1 -1 +1 -1 +1 -1 +1) = 8.$$

Для того чтобы узнать, какой бит послала станция A, осталось нормализовать результат, то есть разделить его на количество разрядов в расширяющей последовательности: 8/8 = 1.

Если бы станция хотела принимать информацию от станции B, то ей нужно было бы при демодуляции использовать код расширения станции B (+1 −1 +1 −1 −1 +1 −1 +1):

$$S \times B = (-2 +4\ 0 +2\ 0 +2 +2\ 0) \times (+1 -1 +1 -1 -1 +1 -1 +1) = -8.$$

После нормализации мы получаем сигнал −1, который соответствует двоичному нулю исходной информации станции B.

Мы объяснили только основную идею CDMA, предельно упростив ситуацию. На практике CDMA является весьма сложной технологией, которая оперирует не условными значениями +1 и −1, а модулированными сигналами, например сигналами BPSK. Кроме того, узлы сети не синхронизированы между собой, а сигналы, которые приходят от удаленных на различные расстояния от приемника узлов, имеют разную мощность. Проблема синхронизации приемника и передатчика решается за счет передачи длинной последовательности определенного кода, называемого **пилотным сигналом**. Для того же, чтобы мощности всех передатчиков были примерно равны для базовой станции, в CDMA применяются специальные процедуры управления мощностью.

Выводы

Беспроводная связь делится на мобильную и фиксированную. Для организации мобильной связи беспроводная среда является единственной альтернативой. Фиксированная беспроводная связь обеспечивает доступ к узлам сети, расположенным в пределах небольшой территории, например здания.

Каждый узел беспроводной линии связи оснащается антенной, которая одновременно является передатчиком и приемником электромагнитных волн.

Электромагнитные волны могут распространяться во всех направлениях или же в пределах определенного сектора. Тип распространения зависит от типа антенны.

Беспроводные системы передачи данных делятся на четыре группы в зависимости от используемого диапазона электромагнитного спектра: широковещательные (радио-) системы, микроволновые системы, системы инфракрасных волн, системы видимого света.

Из-за отражения, дифракции и рассеивания электромагнитных волн возникает многолучевое распространение одного и того же сигнала. Это приводит к межсимвольной интерференции и многолучевому замиранию.

Передача данных в диапазонах 900 МГц, 2,4 ГГц и 5 ГГц, которые получили название ISM-диапазонов, не требует лицензирования, если мощность передатчика не превышает 1 Вт.

Беспроводные двухточечные линии связи служат для создания радиорелейных линий, соединения зданий, а также пары компьютеров.

Беспроводные линии связи с одним источником и несколькими приемниками строятся на основе базовой станции. Такие линии используются в мобильных сотовых сетях, а также в системах фиксированного доступа.

Топология с несколькими источниками и несколькими приемниками характерна для беспроводных локальных сетей.

В системах спутниковой связи используются три группы спутников: геостационарные, среднеорбитальные и низкоорбитальные.

Для кодирования дискретной информации в беспроводных системах прибегают к манипуляции (FSK и PSK) и методам расширения спектра (FHSS и DSSS).

В методах расширения спектра для представления информации используется широкий диапазон частот, это уменьшает влияние на сигналы узкополосных шумов.

На основе методов FHSS и DSSS можно мультиплексировать несколько каналов в одном диапазоне частот. Такая техника мультиплексирования называется множественным доступом с кодовым разделением (CDMA).

Контрольные вопросы

1. В чем достоинства и недостатки беспроводной передачи информации по сравнению с проводной?
2. За счет чего радиоволны с частотами от 2 до 30 МГц могут распространяться на сотни километров?
3. Какие атмосферные явления мешают распространению микроволн?
4. Какие препятствия вызывают дифракцию? Варианты ответов:
   а) непроницаемые препятствия, размер которых соизмерим с длиной волны;
   б) непроницаемые препятствия, размер которых намного больше длины волны;
   в) непроницаемые препятствия, размер которых намного меньше длины волны.
5. Каковы недостатки геостационарного спутника? Варианты ответов:
   а) велики задержки сигнала;
   б) велико затухание сигнала, что приводит к необходимости использования антенн большого диаметра;
   в) мало покрытие территории;
   г) хорошая связь обеспечивается лишь в районах, близких к Северному и Южному полюсам.

ГЛАВА 10 Первичные сети

Назначение и типы первичных сетей

Первичные сети предназначены для создания коммутируемой инфраструктуры, с помощью которой можно достаточно быстро и гибко организовать постоянный канал с двухточечной топологией между двумя пользовательскими устройствами, подключенными к такой сети. Первичные сети также называют транспортными сетями, так как они предоставляют только транспортные услуги, передавая пользовательскую информацию без ее изменения.

На основе каналов, образованных первичными сетями, работают наложенные компьютерные или телефонные сети. В первичных сетях применяется техника коммутации каналов. Каналы, предоставляемые первичными сетями своим пользователям, отличаются высокой пропускной способностью — обычно от 2 Мбит/с до 100 Гбит/с, большой протяженностью, составляющей сотни и тысячи километров, а также высоким качеством, что выражается в очень низком проценте битовых ошибок. Для обеспечения таких скоростей и таких расстояний в наибольшей степени подходит оптоволоконный кабель, поэтому название «оптические сети» прочно закрепилось в качестве еще одного названия первичных сетей, несмотря на то что оптоволоконные кабели применяются и в других типах сетей, например в сетях Ethernet.

Существует несколько поколений технологий первичных сетей:

- **плезиохронная цифровая иерархия** (Plesiochronous Digital Hierarchy, **PDH**);
- **синхронная цифровая иерархия** (Synchronous Digital Hierarchy, **SDH**) — этой технологии в Америке соответствует стандарт SONET;
- **уплотненное волновое мультиплексирование** (Dense Wave Division Multiplexing, **DWDM**);
- **оптические транспортные сети** (Optical Transport Network, **OTN**) — данная технология определят способы передачи данных по волновым каналам DWDM.

В первых двух технологиях (PDH и SDH) для разделения высокоскоростного канала применяется временное мультиплексирование (TDM), а данные передаются в цифровой форме. Каждая из них поддерживает иерархию скоростей, так что пользователь может выбрать подходящую ему скорость для каналов, с помощью которых он будет строить наложенную сеть.

Технология SDH обеспечивает более высокие скорости, чем PDH, так что при построении крупной первичной сети ее магистраль строится на технологии SDH, а сеть доступа — на технологии PDH.

Сети DWDM не являются собственно цифровыми сетями, так как предоставляют своим пользователям выделенную световую волну для передачи информации, которую те могут применять по своему усмотрению и передавать по ней данные как в аналоговой, так и в дискретной (цифровой) форме. Техника мультиплексирования DWDM существенно

повысила пропускную способность современных телекоммуникационных сетей, так как она позволяет организовать в одном оптическом волокне несколько десятков волновых каналов, каждый из которых может переносить информацию независимо от других каналов.

В начальный период развития технологии DWDM волновые каналы использовались в основном для передачи сигналов SDH, то есть мультиплексоры DWDM были одновременно и мультиплексорами SDH для каждого из своих волновых каналов.

Впоследствии для более эффективного использования волновых каналов DWDM была разработана технология OTN, которая позволяет передавать по волновым каналам сигналы любых технологий, включая SDH, Gigabit Ethernet, 10G Ethernet и 100G Ethernet.

Сети PDH

Технология плезиохронной цифровой иерархии (PDH) была разработана в конце 60-х годов компанией AT&T для решения проблемы связи крупных коммутаторов телефонных сетей между собой. Линии связи FDM, применяемые ранее для решения этой задачи и поддерживающие передачу голоса в аналоговой форме, исчерпали свои возможности в плане организации высокоскоростной многоканальной связи по одному кабелю. В технологии FDM по витой медной паре одновременно передавалось только 12 абонентских каналов, так что для передачи между телефонными станциями большего количества абонентских каналов приходилось прокладывать кабели с увеличенным количеством пар проводов или более дорогие коаксиальные кабели.

Иерархия скоростей

Начало технологии PDH было положено разработкой мультиплексора **T-1**, который позволял в цифровом виде мультиплексировать, передавать и коммутировать (на постоянной основе) голосовой трафик 24 абонентов. Так как абоненты по-прежнему пользовались обычными телефонными аппаратами, то есть передача голоса шла в аналоговой форме, то мультиплексоры T-1 сами осуществляли оцифровывание голоса с частотой 8000 Гц и кодировали голос методом импульсно-кодовой модуляции. В результате каждый абонентский канал образовывал цифровой поток данных 64 Кбит/с, а мультиплексор T-1 обеспечивал передачу на скорости 1,544 Мбит/с.

В качестве средств мультиплексирования при соединении крупных телефонных станций каналы T-1 имели недостаточную пропускную способность, поэтому была реализована идея образования каналов с *иерархией скоростей*. Четыре канала типа T-1 объединили в канал следующего уровня цифровой иерархии — T-2, передающий данные со скоростью 6,312 Мбит/с. Канал T-3, образованный путем объединения семи каналов T-2, имеет скорость 44,736 Мбит/с. Канал T-4 объединяет 6 каналов T-3, в результате его скорость равна 274 Мбит/с. Описанная технология получила название **системы T-каналов**.

С середины 70-х годов выделенные каналы, построенные на основе систем T-каналов, стали сдаваться телефонными компаниями в аренду на коммерческих условиях, перестав быть внутренней технологией этих компаний. Системы T-каналов позволяют передавать не только голос, но и любые данные, представленные в цифровой форме: компьютерные данные, телевизионное изображение, факсы и т. п.

Технология систем Т-каналов была стандартизована Американским национальным институтом стандартов (ANSI), а позже — международной организацией ITU-T. При стандартизации она получила название плезиохронной цифровой иерархии (PDH). В результате внесенных ITU-T изменений возникла несовместимость американской и международной версий стандарта PDH. Аналогом систем Т-каналов в международном стандарте являются каналы типа **E-1**, **E-2** и **E-3** с отличающимися скоростями — соответственно 2,048, 8,488 и 34,368 Мбит/с. Американская версия сегодня помимо США распространена также в Канаде и Японии (с некоторыми различиями), в Европе же применяется международный стандарт ITU-T.

Несмотря на различия, в американской и международной версиях технологии цифровой иерархии принято использовать одни и те же обозначения для иерархии скоростей — DSn (Digital Signal n). В табл. 10.1 приводятся значения для всех введенных стандартами уровней скоростей обеих технологий.

Таблица 10.1. Иерархия цифровых скоростей

Америка				**ITU-T (Европа)**		
обозначение скорости	**количество голосовых каналов**	**количество каналов предыдущего уровня**	**скорость, Мбит/с**	**количество голосовых каналов**	**количество каналов предыдущего уровня**	**скорость, Мбит/с**
DS-0	1	1	64 Кбит/с	1	1	64 Кбит/с
DS-1	24	24	1,544	30	30	2,048
DS-2	96	4	6,312	120	4	8,488
DS-3	672	7	44,736	480	4	34,368
DS-4	4032	6	274,176	1920	4	139,264

На практике в основном используются каналы Т-1/Е-1 и Т-3/Е-3.

Методы мультиплексирования

Мультиплексор Т-1 обеспечивает передачу данных 24 абонентов со скоростью 1,544 Мбит/с в кадре, имеющем достаточно простой формат. В этом кадре последовательно передается по одному байту каждого абонента, а после 24 байт вставляется один *бит синхронизации*. Первоначально устройства Т-1 функционировали только на внутренних тактовых генераторах и каждый кадр с помощью битов синхронизации мог передаваться асинхронно.

> Сегодня мультиплексоры и коммутаторы первичной сети PDH работают на централизованной тактовой частоте, распределяемой из одной или нескольких точек сети.

Однако принцип формирования кадра не изменился, поэтому биты синхронизации в кадре по-прежнему присутствуют. Суммарная скорость пользовательских каналов составляет $24 \times 64 = 1{,}536$ Мбит/с, а еще 8 Кбит/с добавляют биты синхронизации, итого получается 1,544 Мбит/с.

Теперь рассмотрим еще одну особенность формата кадра T-1. В аппаратуре T-1 восьмой бит каждого байта в кадре имеет назначение, зависящее от типа передаваемых данных и поколения аппаратуры. При передаче *голоса* с помощью этого бита переносится служебная информация, к которой относятся номер вызываемого абонента и другие сведения, необходимые для установления соединения между абонентами сети. Протокол, обеспечивающий такое соединение, называется в телефонии **сигнальным протоколом**. Поэтому реальная скорость передачи пользовательских данных в этом случае составляет не 64, а 56 Кбит/с. Техника применения восьмого бита для служебных целей получила название **«кражи бита»**.

Версия технологии PDH, описанная в международных стандартах G.700–G.706 ITU-T, как уже отмечалось, имеет отличия от американской технологии систем T-каналов. В частности, в ней не используется схема «кражи бита». При переходе к следующему уровню иерархии коэффициент кратности скорости имеет постоянное значение 4. Вместо восьмого бита в канале E-1 на служебные цели отводятся 2 байта из 32, а именно нулевой (для целей синхронизации приемника и передатчика) и шестнадцатый (в нем передается служебная сигнальная информация). Для голосовых или компьютерных данных остается 30 каналов со скоростью передачи 64 Кбит/с каждый.

При мультиплексировании нескольких пользовательских потоков в мультиплексорах PDH применяется техника, называемая **бит-стаффингом**. К этой технике прибегают, когда скорость пользовательского потока оказывается несколько меньше, чем скорость объединенного потока, — подобные проблемы могут возникать в сети, состоящей из большого количества мультиплексоров, несмотря на все усилия по централизованной синхронизации узлов сети (в природе нет ничего идеального, в том числе идеально синхронных узлов сети). В результате мультиплексор PDH периодически сталкивается с ситуацией, когда ему «не хватает» бита для представления в объединенном потоке того или иного пользовательского потока. В этом случае мультиплексор просто вставляет в объединенный поток бит-вставку и отмечает этот факт в служебный битах объединенного кадра. При демультиплексировании объединенного потока бит-вставка удаляется из пользовательского потока, который возвращается в исходное состояние. Техника бит-стаффинга применяется как в международной, так и в американской версии PDH.

Отсутствие полной синхронности потоков данных при объединении низкоскоростных каналов в высокоскоростные и дало название технологии PDH («плезиохронный» означает «почти синхронный»).

Синхронизация сетей PDH

В случае небольшой сети PDH, например сети города, синхронизация всех устройств сети из одной точки представляется достаточно простым делом. Однако для более крупных сетей, например сетей масштаба страны, которые состоят из некоторого количества региональных сетей, синхронизация всех устройств сети представляет собой проблему.

Общий подход к решению этой проблемы описан в стандарте ITU-T G.810. Он заключается в организации в сети иерархии эталонных источников синхросигналов, а также системы распределения синхросигналов по всем узлам сети (рис. 10.1).

Каждая крупная сеть должна иметь по крайней мере один **первичный эталонный генератор** (ПЭГ) синхросигналов (в англоязычном варианте — Primary Reference Clock, PRC).

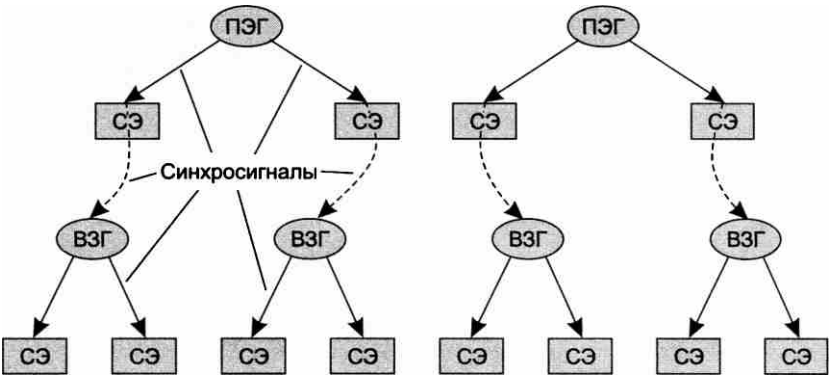


Рис. 10.1. Организация распределения синхросигналов по узлам сети PDH

Это очень точный источник синхросигналов, способный вырабатывать синхросигналы с относительной точностью частоты не хуже 10^{-11} (такую точность требуют стандарты ITU-T G.811 и ANSI T1.101, в последнем для описания точности ПЭГ применяется название **Stratum 1**). На практике в качестве ПЭГ используют либо автономные атомные (водородные или цезиевые) часы, либо часы, синхронизирующиеся от спутниковых систем точного мирового времени, таких как GPS или ГЛОНАСС. Обычно точность ПЭГ достигает 10^{-13}.

Стандартным синхросигналом является сигнал тактовой частоты уровня DS1, то есть частоты 2048 кГц для международного варианта стандартов PDH и 1544 кГц для американского варианта этих стандартов.

Синхросигналы от ПЭГ непосредственно поступают на специально отведенные для этой цели синхровходы магистральных устройств сети PDH. В том случае, если это составная сеть, то каждая крупная сеть, входящая в состав составной сети (например, региональная сеть, входящая в состав национальной сети), имеет свой ПЭГ.

Для синхронизации немагистральных узлов используется **вторичный задающий генератор (ВЗГ)** синхросигналов, который в варианте ITU-T называют Secondary Reference Clock (SRC), а в варианте ANSI — генератором уровня **Stratum 2**. ВЗГ работает в режиме принудительной синхронизации, являясь ведомым таймером в паре ПЭГ-ВЗГ. Обычно ВЗГ получает синхросигналы от некоторого ПЗГ через промежуточные магистральные узлы сети, при этом для передачи синхросигналов используются биты служебных байтов кадра, например нулевого байта кадра E-1 в международном варианте PDH.

Точность ВЗГ меньше, чем точность ПЭГ: ITU-T в стандарте G.812 определяет ее как «не хуже 10^{-9}», а точность генераторов Stratum 2 должна быть не «хуже $1{,}6 \times 10^{-8}$».

Иерархия эталонных генераторов может быть продолжена, если это необходимо, при этом точность каждого более низкого уровня, естественно, понижается. Генераторы нижних уровней могут использовать для выработки своих синхросигналов несколько эталонных генераторов более высокого уровня, но при этом в каждый момент времени один из них должен быть основным, а остальные — резервными; такое построение системы синхронизации обеспечивает ее отказоустойчивость. Однако в этом случае нужно приоритезировать сигналы генераторов более высоких уровней. Кроме того, при построении системы синхронизации требуется гарантировать отсутствие петель синхронизации.

Методы синхронизации цифровых сетей, кратко описанные в этом разделе, применимы не только к сетям PDH, но и к другим сетям, работающим на основе синхронного TDM-мультиплексирования, например к сетям SDH, а также к сетям цифровых телефонных коммутаторов.

Сети SONET/SDH

Как американский, так и международный вариант технологии PDH обладает рядом недостатков, основным из которых является сложность и неэффективность операций мультиплексирования и демультиплексирования пользовательских данных. Применение техники бит-стаффинга для выравнивания скоростей потоков приводит к тому, что для извлечения пользовательских данных одного из каналов объединенного канала необходимо полностью (!) демультиплексировать кадры объединенного канала.

Также в технологии PDH не предусмотрены встроенные средства обеспечения отказоустойчивости и администрирования сети.

Наконец, недостатком PDH являются слишком низкие по современным понятиям скорости передачи данных — ее иерархия скоростей заканчивается уровнем 139 Мбит/с.

Недостатки и ограничения технологии PDH были учтены и преодолены разработчиками технологии **синхронных оптических сетей** (Synchronous Optical NET, SONET), первый вариант стандарта которой появился в 1984 году. Затем она была стандартизована комитетом T-1 института ANSI. Международная стандартизация технологии проходила под эгидой Европейского института телекоммуникационных стандартов (European Telecommunications Standards Institute, ETSI) и сектором телекоммуникационной стандартизации союза ITU (ITU Telecommunication Standardization Sector, ITU-T) совместно с ANSI и ведущими телекоммуникационными компаниями Америки, Европы и Японии.

Основной целью разработчиков международного стандарта было создание технологии, способной передавать трафик всех существующих цифровых каналов уровня PDH (как американских T1–T3, так и европейских E1–E4) по высокоскоростной магистральной сети на базе волоконно-оптических кабелей и обеспечить иерархию скоростей, продолжающую иерархию технологии PDH до скорости в несколько гигабит в секунду.

В результате длительной работы ITU-T и ETSI удалось разработать и принять международный стандарт **SDH** (Synchronous Digital Hierarchy — синхронная цифровая иерархия). Кроме того, стандарт SONET был доработан так, чтобы аппаратура и сети SDH и SONET являлись совместимыми и могли мультиплексировать входные потоки практически любого стандарта PDH — и американского, и европейского.

Иерархия скоростей и методы мультиплексирования

Поддерживаемая технологией SONET/SDH иерархия скоростей представлена в табл. 10.2.

В стандарте SDH все уровни скоростей (и соответственно форматы кадров для этих уровней) имеют общее название STM-N (Synchronous Transport Module level N — синхронный транспортный модуль уровня N). В технологии SONET существуют два обозначения для уровней скоростей: название STS-N (Synchronous Transport Signal level N — синхронный

транспортный сигнал уровня N) употребляется в случае передачи данных электрическим сигналом, а название OC-N (Optical Carrier level N — оптоволоконная линия связи уровня N) используют в случае передачи данных по волоконно-оптическому кабелю. Далее для упрощения изложения мы сосредоточимся на технологии SDH.

Таблица 10.2. Иерархия скоростей SONET/SDH

SDH	SONET	Скорость
	STS-1, OC-1	51,84 Мбит/с
STM-1	STS-3, OC-3	155,520 Мбит/с
STM-3	OC-9	466,560 Мбит/с
STM-4	OC-12	622,080 Мбит/с
STM-6	OC-18	933,120 Мбит/с
STM-8	OC-24	1,244 Гбит/с
STM-12	OC-36	1,866 Гбит/с
STM-16	OC-48	2,488 Гбит/с
STM-64	OC-192	9,953 Гбит/с
STM-256	OC-768	39,81 Гбит/с

Кадры STM-N имеют достаточно сложную структуру, позволяющую агрегировать в общий магистральный поток потоки SDH и PDH различных скоростей, а также выполнять операции ввода-вывода без полного демультиплексирования магистрального потока.

Операции мультиплексирования и ввода-вывода выполняются при помощи **виртуальных контейнеров** (Virtual Container, VC), в которых блоки данных PDH можно транспортировать через сеть SDH. Помимо блоков данных PDH в виртуальный контейнер помещается еще некоторая служебная информация, в частности **заголовок пути** (Path OverHead, POH) контейнера, в котором размещается статистическая информация о процессе прохождения контейнера вдоль пути от его начальной до конечной точки (сообщения об ошибках), а также другие служебные данные, например индикатор установления соединения между конечными точками. В результате размер виртуального контейнера оказывается больше, чем соответствующая нагрузка в виде блоков данных PDH, которую он переносит. Например, виртуальный контейнер VC-12 помимо 32 байт данных потока E-1 содержит еще 3 байта служебной информации.

В технологии SDH определено несколько типов виртуальных контейнеров (рис. 10.2), предназначенных для транспортировки основных типов блоков данных PDH: VC-11 (1,5 Мбит/с), VC-12 (2 Мбит/с), VC-2 (6 Мбит/с), VC3 (34/45 Мбит/с) и VC-4 (140 Мбит/с).

Виртуальные контейнеры являются *единицей коммутации* мультиплексоров SDH. В каждом мультиплексоре существует **таблица соединений** (называемая также **таблицей кросс-соединений**), в которой указано, например, что контейнер VC-12 порта P1 соединен с контейнером VC12 порта P5, а контейнер VC3 порта P8 — с контейнером VC3 порта P9. Таблицу соединений формирует администратор сети с помощью системы управления или управляющего терминала на каждом мультиплексоре так, чтобы обеспечить сквозной путь между конечными точками сети, к которым подключено пользовательское оборудование.

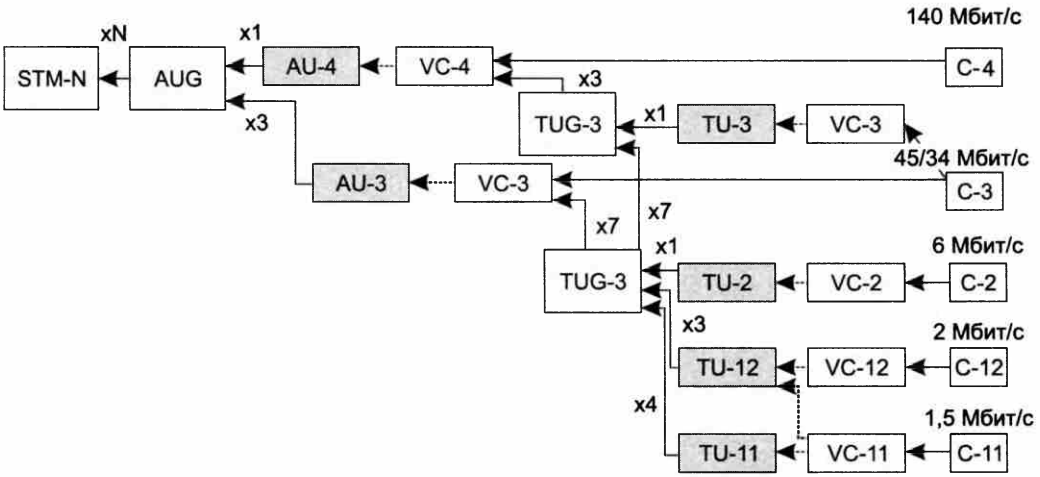


Рис. 10.2. Схема мультиплексирования данных в SDH

Чтобы совместить в рамках одной сети механизмы синхронной передачи кадров (STM-N) и асинхронный характер переносимых этими кадрами пользовательских данных PDH, в технологии SDH применяются **указатели**. Концепция указателей — ключевая в технологии SDH, она заменяет принятое в PDH выравнивание скоростей асинхронных источников посредством дополнительных битов. Указатель определяет текущее положение виртуального контейнера в агрегированной структуре более высокого уровня, каковой является **трибутариый блок** (Tributary Unit, TU) либо **административный блок** (Administrative Unit, AU). Собственно, основное отличие этих блоков от виртуального контейнера заключается в наличии дополнительного поля указателя. С помощью этого указателя виртуальный контейнер может «смещаться» в определенных пределах внутри своего трибутарного или административного блока, если скорость пользовательского потока несколько отличается от скорости кадра SDH, куда этот поток мультиплексируется.

> Именно благодаря системе указателей мультиплексор находит положение пользовательских данных в синхронном потоке байтов кадров STM-N и «на лету» извлекает их оттуда, чего механизм мультиплексирования, применяемый в PDH, делать не позволяет.

Трибутарные блоки объединяются в группы, а те, в свою очередь, входят в административные блоки. Группа административных блоков (Administrative Unit Group, AUG) в количестве N и образует полезную нагрузку кадра STM-N. Помимо этого, в кадре имеется заголовок с общей для всех блоков AU служебной информацией. На каждом шаге преобразования к предыдущим данным добавляется несколько служебных байтов: они помогают распознать структуру блока или группы блоков и затем определить с помощью указателей начало пользовательских данных.

На рис. 10.2 структурные единицы кадра SDH, содержащие указатели, заштрихованы, а связь между контейнерами и блоками, допускающая сдвиг данных по фазе, показана пунктиром.

Схема мультиплексирования SDH предоставляет разнообразные возможности по объединению пользовательских потоков PDH. Например, для кадра STM-1 можно реализовать такие варианты:

- 1 поток E-4;
- 63 потока E-1;
- 1 поток E-3 и 42 потока E-1.

Другие варианты читатель может предложить сам.

В технологии SDH также применяется описанная в главе 8 техника *прямой коррекции ошибок* (FEC). Напомним, что эта техника основана на применении самокорректирующих кодов, позволяющих исправлять искажения битов данных «на лету», то есть не прибегая к их повторной передаче, а используя избыточную часть кода. Такая техника может существенно повысить эффективную скорость передачи данных при наличии помех или сбоев в работе приемопередатчиков. Обычно к прямой коррекции ошибок мультиплексоры SDH прибегают на скоростях 2,5 Гбит/с и выше.

Типы оборудования

Основным элементом сети SDH является **мультиплексор** (рис. 10.3). Обычно он оснащен некоторым количеством портов PDH и SDH: например, портами PDH на 2 и 34/45 Мбит/с и портами SDH STM-1 на 155 Мбит/с и STM-4 на 622 Мбит/с. Порты мультиплексора SDH делятся на агрегатные и трибутарные.

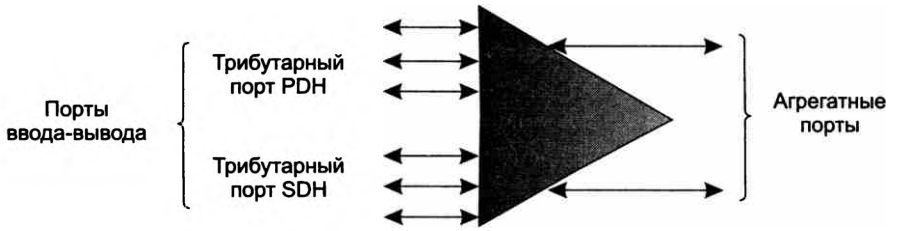


Рис. 10.3. Мультиплексор SDH

Трибутарные порты часто называют также портами ввода-вывода, а **агрегатные** — линейными портами. Эта терминология отражает типовые топологии сетей SDH, где имеется ярко выраженная магистраль в виде цепи или кольца, по которой передаются потоки данных, поступающие от пользователей сети через порты ввода-вывода (трибутарные порты), то есть втекающие в агрегированный поток («tributary» дословно означает «приток»).

Мультиплексоры SDH обычно разделяют на два типа, разница между которыми определяется положением мультиплексора в сети SDH (рис. 10.4).

- **Терминальный мультиплексор** (Terminal Multiplexer, TM) *завершает* агрегатный канал, мультиплексируя в нем большое количество трибутарных каналов, поэтому он оснащен одним агрегатным и множеством трибутарных портов.
- **Мультиплексор ввода-вывода** (Add-Drop Multiplexer, ADM занимает промежуточное положение на магистрали (в кольце, цепи или смешанной топологии). Он имеет два

агрегатных порта, транзитом передавая агрегатный поток данных. С помощью небольшого количества трибутарных портов такой мультиплексор вводит в агрегатный поток или выводит из агрегатного потока данные трибутарных каналов.

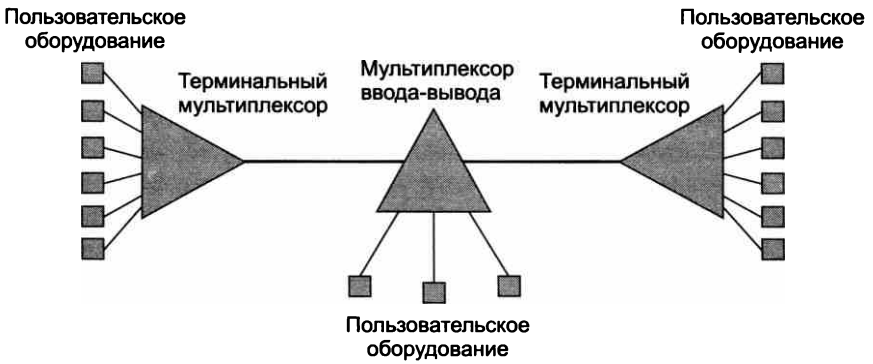


Рис. 10.4. Типы мультиплексоров SDH

Иногда также выделяют мультиплексоры, которые выполняют операции коммутации над произвольными виртуальными контейнерами — так называемые **цифровые кросс-коннекторы** (Digital Cross-Connect, DXC). В таких мультиплексорах не делается различий между агрегатными и трибутарными портами, так как они предназначены для работы в ячеистой топологии, где выделить агрегатные потоки невозможно.

Помимо мультиплексоров в состав сети SDH могут входить **регенераторы сигналов**, необходимые для преодоления ограничений по расстоянию между мультиплексорами. Эти ограничения зависят от мощности оптических передатчиков, чувствительности приемников и затухания волоконно-оптического кабеля. Регенератор преобразует оптический сигнал в электрический и обратно, при этом восстанавливается форма сигнала и его временны́е характеристики. В настоящее время регенераторы SDH применяются достаточно редко, так как стоимость их ненамного ниже стоимости мультиплексора, а функциональные возможности несоизмеримо беднее.

Типовые топологии

В сетях SDH применяются различные топологии связей. Наиболее часто используются кольца и линейные цепи мультиплексоров, также находит все большее применение ячеистая топология, близкая к полносвязной.

Кольцо SDH строится из мультиплексоров ввода-вывода, имеющих по крайней мере по два агрегатных порта (рис. 10.5, *а*). Пользовательские потоки вводятся в кольцо и выводятся из кольца через трибутарные порты, образуя двухточечные соединения (на рисунке показаны в качестве примера два таких соединения). Кольцо является классической регулярной топологией, обладающей потенциальной отказоустойчивостью — при однократном обрыве кабеля или выходе из строя мультиплексора соединение сохранится, если его направить по кольцу в противоположном направлении. Кольцо обычно строится на основе кабеля с двумя оптическими волокнами, но иногда для повышения надежности и пропускной способности применяют четыре волокна.

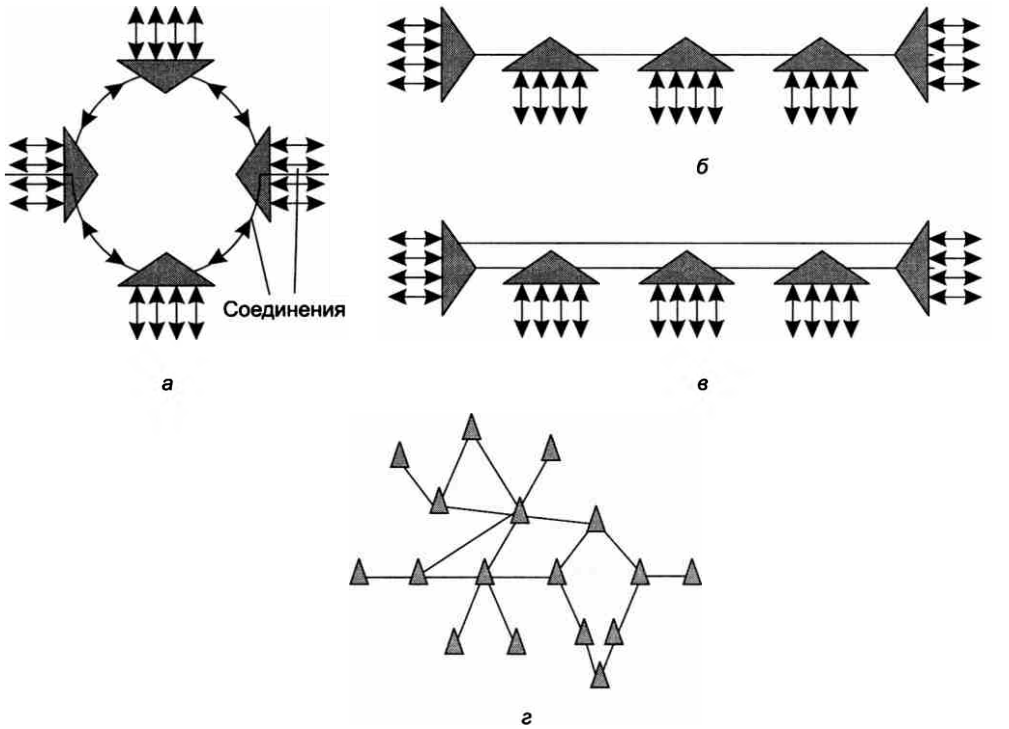


Рис. 10.5. Типовые топологии

Цепь (рис. 10.5, *б*) — это линейная последовательность мультиплексоров, из которых два оконечных играют роль терминальных мультиплексоров, остальные — мультиплексоров ввода-вывода. Обычно сеть с топологией цепи применяется в тех случаях, когда узлы имеют соответствующее географическое расположение, например вдоль магистрали железной дороги или трубопровода. Правда, в подобных случаях может применяться и **плоское кольцо** (рис. 10.5, *в*), обеспечивающее более высокий уровень отказоустойчивости за счет двух дополнительных волокон в магистральном кабеле и по одному дополнительному агрегатному порту у терминальных мультиплексоров.

Эти базовые топологии могут комбинироваться при построении сложной и разветвленной сети SDH, образуя участки с радиально-кольцевой топологией, соединениями «кольцо-кольцо» и т. п. Наиболее общим случаем является **ячеистая топология** (рис. 10.5, *г*), в которой мультиплексоры соединяются друг с другом большим количеством связей, за счет чего сеть может достичь очень высокой степени производительности и надежности.

Методы обеспечения живучести сети

Одной из сильных сторон первичных сетей SDH является разнообразный набор средств отказоустойчивости, который позволяет сети быстро (за десятки миллисекунд) восстановить работоспособность в случае отказа какого-либо элемента сети — линии связи, порта или карты мультиплексора либо мультиплексора в целом.

В SDH в качестве общего названия механизмов отказоустойчивости используется термин **автоматическое защитное переключение** (Automatic Protection Switching, APS), отражающий факт перехода (переключения) на резервный путь или резервный элемент мультиплексора при отказе основного. Сети, поддерживающие такой механизм, в стандартах SDH названы **самовосстанавливающимися**.

В сетях SDH применяются три схемы защиты.

- **Защита 1 + 1** означает, что резервный элемент выполняет ту же работу, что и основной. Например, при защите трибутарной карты по схеме 1 + 1 трафик проходит как через рабочую карту (резервируемую), так и через защитную (резервную).
- **Защита 1 : 1** подразумевает, что защитный элемент в нормальном режиме не выполняет функции защищаемого элемента, а переключается на них только в случае отказа.
- **Защита 1 : N** предусматривает выделение одного защитного элемента на *N* защищаемых. При отказе одного из защищаемых элементов его функции начинает выполнять защитный, при этом остальные элементы остаются без защиты — до тех пор, пока отказавший элемент не будет заменен.

В зависимости от типа защищаемого путем резервирования элемента сети в оборудовании и сетях SDH применяются следующие основные виды автоматической защиты: защита мультиплексной секции, защита сетевого соединения, разделяемая защита мультиплексной секции в кольцевой топологии.

Защита мультиплексной секции (Multiplex Section Protection, MSP) работает между двумя смежными мультиплексорами, она включает две пары портов и две линии связи (возможно, в свою очередь, включающие регенераторы, но не мультиплексоры). Обычно применяется схема защиты 1 + 1. При этом для рабочего канала (верхняя пара соединенных кабелем портов на рис. 10.6, *а*) конфигурируется защитный канал (нижняя пара портов). При установлении защиты MSP в каждом мультиплексоре необходимо выполнить конфигурирование, указав связь между рабочим и защитным портами. В исходном состоянии весь трафик передается по обоим каналам (как по рабочему, так и по защитному).

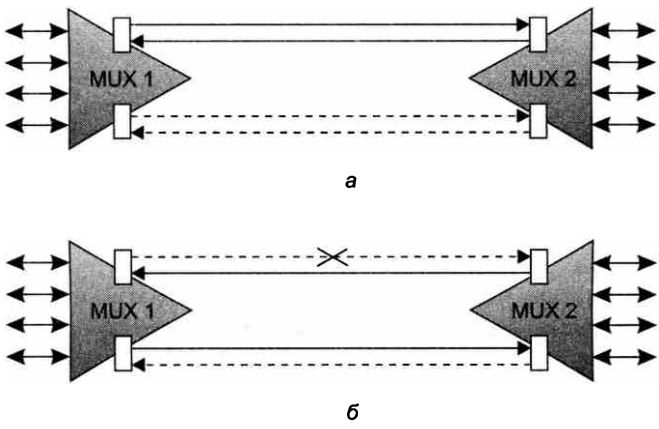


Рис. 10.6. Защита мультиплексной секции

Существует однонаправленная и двунаправленная защита MSP. При *однонаправленной* защите переключение выполняет только тот мультиплексор, который обнаруживает отсутствие или искажение принимаемых данных по рабочему каналу (из-за отказа порта, ошибки сигнала, деградации сигнала и т. п.). После этого данный мультиплексор переходит на прием данных по защитному каналу. Если в обратном направлении рабочий канал продолжает нормально передавать данные, то второй мультиплексор продолжает его использовать для передачи данных (именно этот случай показан на рис. 10.6, *б*).

В случае *двунаправленной* защиты MSP при отказе рабочего канала в каком-либо направлении выполняется полное переключение на защитные порты мультиплексоров. Для уведомления передающего мультиплексора о необходимости переключения принимающий мультиплексор, обнаружив отказ, использует протокол, называемый протоколом «K-байт». Уведомление происходит по рабочему каналу за счет помещения в два байта заголовка кадра STM-N информации о статусах рабочего и защитного каналов, а также о типе отказа. Механизм MSP обеспечивает защиту всех соединений, проходящих через защищаемую мультиплексную секцию. Время переключения защиты MSP, согласно требованиям стандарта, не должно превышать 50 мс.

Защита сетевого соединения (Sub-Network Connection Protection, SNC-P), то есть защита пути (соединения) через сеть для определенного виртуального контейнера, обеспечивает переключение требуемого пользовательского соединения на альтернативный путь при отказе основного пути. Объектом защиты SNC-P является трибутарный трафик, помещенный в виртуальный контейнер заданного типа (например, в VC12, VC-3 или VC-4). Используется схема защиты 1 + 1.

Защита SNC-P конфигурируется в двух мультиплексорах: во входном, в котором трибутарный трафик, помещенный в виртуальный контейнер, разветвляется, и в выходном, в котором сходятся два альтернативных пути трафика. Пример защиты SNC-P показан на рис. 10.7. В мультиплексоре ADM1 для виртуального контейнера VC-4 трибутарного порта T-2 заданы два соединения: с одним из четырех контейнеров VC-4 агрегатного порта A1 и с одним из четырех контейнеров VC-4 агрегатного порта A2. Одно из соединений конфигурируется как рабочее, второе — как защитное, при этом трафик передается по обоим соединениям. Промежуточные (для данных соединений) мультиплексоры конфигурируются обычным образом. В выходном мультиплексоре контейнер VC-4 трибутарного порта T-3 также соединяется с контейнерами — агрегатного порта A1 и агрегатного порта A2. Из двух поступающих на порт T3 потоков выбирается тот, качество которого выше (при равном нормальном качестве выбирается сигнал из агрегатного порта, указанного при конфигурировании в качестве рабочего).

Защита SNC-P работает в любых топологиях сетей SDH, в которых имеются альтернативные пути следования трафика, то есть кольцевых и ячеистых.

Разделяемая защита мультиплексной секции в кольцевой топологии (Multiplex Section Shared Protection Ring, MS-SPRing) обеспечивает в некоторых случаях более экономичную защиту трафика в кольце по сравнению с защитой SNC-P за счет того, что полоса пропускания не резервируется заранее для каждого соединения. Вместо этого резервируется половина пропускной способности кольца, но эта резервная полоса выделяется для соединений динамически, по мере необходимости, то есть после обнаружения факта отказа линии или мультиплексора. Степень экономии полосы при применении защиты MS-SPRing зависит от распределения трафика.

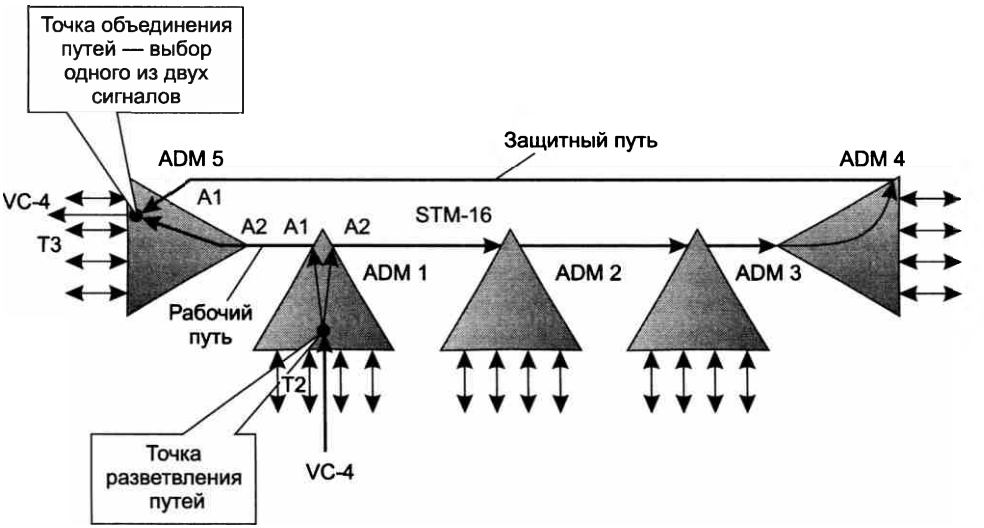


Рис. 10.7. Защита сетевого соединения

Новое поколение протоколов SDH

Изначально технология SDH была ориентирована на передачу элементарных потоков голосового трафика, отсюда и ее ориентация на мультиплексирование пользовательских потоков со скоростями, кратными 64 Кбит/с, и применение коэффициента кратности 4 для иерархии скоростей.

Однако популярность Интернета изменила ситуацию в телекоммуникационном мире, и сегодня объемы компьютерного трафика в первичных сетях превосходят объемы голосового трафика. В условиях доминирования Ethernet как технологии канального уровня почти весь компьютерный трафик, поступающий на входы мультиплексоров первичных сетей, представляет собой кадры Ethernet, а значит, представлен иерархией скоростей 10–100 Мбит/с и 1–10–100 Гбит/с. Пользовательские потоки с такими скоростями не очень эффективно укладываются в виртуальные контейнеры SDH, рассчитанные на решение других задач.

Для исправления ситуации организация ITU-T разработала несколько стандартов, которые составляют так называемую технологию SDH нового поколения (SDH Next Generation, или SDH NG). Эти стандарты делают технологию SDH более дружественной к компьютерным данным.

Стандарты SDH нового поколения описывают три новых механизма:

- виртуальная конкатенация (VCAT);
- схема динамического изменения пропускной способности линии (LCAS);
- общая процедура инкапсуляции (кадрирования) данных (GFP).

Виртуальная конкатенация (Virtual Concatenation, VCAT) контейнеров позволяет более эффективно использовать емкость виртуальных контейнеров SDH при передаче трафика Ethernet.

Виртуальная конкатенация позволяет объединять несколько виртуальных контейнеров в один виртуальный конкатенированный контейнер. При этом объединяемые контейнеры должны быть одного типа, например только VC-3 или только VC-12.

Коэффициент кратности при объединении может быть любым, от 1 до максимального числа, определяемого емкостью кадра STM-N, применяемого для передачи объединенного контейнера. При виртуальной конкатенации объединенный контейнер обозначается как VC-*N*-*M*v, где *N* — тип виртуального контейнера, а *M* — кратность его использования, например VC-3-21v.

Название «виртуальная конкатенация» отражает тот факт, что только конечные мультиплексоры (то есть тот мультиплексор, который формирует объединенный контейнер из пользовательских потоков, и тот мультиплексор, который его демультиплексирует в пользовательские потоки) должны понимать, что это конкатенированный контейнер. Все промежуточные мультиплексоры сети SDH рассматривают составляющие виртуальные контейнеры как независимые и могут передавать их к конечному мультиплексору по разным маршрутам. Конечный мультиплексор выдерживает некоторый тайм-аут перед демультиплексированием пользовательских потоков, что может потребоваться для прибытия всех составляющих контейнеров в том случае, когда они передаются по разным маршрутам.

Посмотрим, как виртуальная конкатенация повышает эффективность передачи трафика Ethernet. Например, чтобы передавать один поток Fast Ethernet 100Мбит/с, в сети STM-16 можно применить виртуальную конкатенацию контейнеров VC-12. Этот тип контейнера обеспечивает передачу пользовательских данных со скоростью 2,176 Мбит/с, поэтому, объединяя 46 таких контейнеров, то есть применяя виртуальную конкатенацию VC12-46v, мы создаем канал с пропускной способностью 100,096 Мбит/с, то есть мы расходуем пропускную способность сети SDH очень эффективно. Оставшиеся 206 контейнеров VC-12 (кадр STM-4 вмещает $63 \times 4 = 252$ контейнера VC-12) можно использовать как для передачи других потоков Fast Ethernet, так и для передачи голосового трафика.

Схема динамического изменения пропускной способности линии (Link Capacity Adjustment Scheme, LCAS) является дополнением к механизму виртуальной конкатенации. Эта схема позволяет исходному мультиплексору, то есть тому, который формирует объединенный контейнер, динамически изменять его емкость, присоединяя к нему или отсоединяя от него виртуальные контейнеры. Для того чтобы добиться нужного эффекта, исходный мультиплексор посылает конечному мультиплексору специальное служебное сообщение, уведомляющее об изменении состава объединенного контейнера.

Общая процедура инкапсуляции данных (Generic Framing Procedure, GFP) предназначена для упаковки кадров различных протоколов компьютерных сетей в кадр единого формата и передачи его по сети SDH. Такая процедура полезна, так как она решает несколько задач, общих при передаче данных компьютерных сетей через сети SDH. В эти задачи входят выравнивание скорости компьютерного протокола со скоростью виртуального контейнера SDH, используемого для передачи компьютерных данных, а также распознавание начала кадра.

- *Выравнивание скорости компьютерного протокола и скорости виртуального контейнера SDH, используемого для передачи компьютерных данных.* Например, если мы применяем объединенный контейнер VC-12-46v для передачи кадров Fast Ethernet, то нужно выровнять скорости 100 и 100,096 Мбит/с. Процедура GFP поддерживает два режима работы: **GFP-F** (кадровый режим, или Frame Mode) и **GFP-T** (прозрачный режим, или

Transparent Mode). В режиме GFP-F проблема выравнивания скоростей решается обычным для компьютерных сетей способом — поступающий кадр полностью буферизуется, упаковывается в формат GFP, а затем со скоростью соединения SDH передается через сеть. Режим GFP-T предназначен для чувствительного к задержкам трафика, в этом режиме кадр полностью не буферизуется, а побитно по мере поступления передается в сеть SDH (предварительно снабженный служебными полями GFP). Для выравнивания скоростей в режиме GFP-T применяются специальные служебные «пустые» кадры GFP, которые посылаются в те моменты, когда рассогласование приводит к отсутствию пользовательских битов у исходного мультиплексора SDH. В нашем примере в режиме GFP-T такие кадры будут посылаться, так как скорость сети SDH чуть выше, чем скорость поступления данных от клиента Fast Ethernet.

- *Распознавание начала кадра.* Соединение SDH представляет для пользователя поток битов, разбитый на кадры SDH, начало которых никак не связано с началом кадра пользователя. Процедура GFP позволяет принимающему мультиплексору SDH распознать начало каждого пользовательского кадра, что необходимо для его извлечения из потока битов, проверки его корректности и передачи на выходной интерфейс в сеть пользователя. В процедуре GFP для распознавания начала кадра служит его собственный заголовок, который состоит из поля длины размером в два байта и поля контрольной суммы поля длины также размером в два байта.

Сети DWDM

Технология **уплотненного волнового мультиплексирования** (Dense Wave Division Multiplexing, DWDM) предназначена для создания оптических магистралей нового поколения, работающих на мультигигабитных и терабитных скоростях. Такой революционный скачок производительности обеспечивает принципиально иной, нежели у SDH, метод мультиплексирования — информация в оптическом волокне передается одновременно большим количеством световых волн — **лямбд** — термин возник в связи с традиционным для физики обозначением длины волны λ.

Сети DWDM работают по принципу коммутации каналов, при этом каждая световая волна представляет собой отдельный *спектральный канал* и несет собственную информацию.

Оборудование DWDM не занимается непосредственно проблемами передачи данных на каждой волне, то есть способом кодирования информации и протоколом ее передачи. Его основными функциями являются операции *мультиплексирования* и *демультиплексирования*, а именно — объединение различных волн в одном световом пучке и выделение информации каждого спектрального канала из общего сигнала. Наиболее развитые устройства DWDM могут также *коммутировать* волны.

ВНИМАНИЕ

Технология DWDM является революционной не только потому, что она в десятки раз повысила верхний предел скорости передачи данных по оптическому волокну, но и потому, что открыла новую эру в технике мультиплексирования и коммутации, выполняя эти операции над световыми сигналами без преобразования их в электрическую форму. Во всех других технологиях, в которых световые сигналы также используются для передачи информации по оптическим волокнам, например SDH и Gigabit Ethernet, световые сигналы обязательно преобразуются в электрические и только потом их можно мультиплексировать и коммутировать.

Принципы работы

Сегодня оборудование DWDM позволяет передавать по одному оптическому волокну до 80 волн разной длины в окне прозрачности 1550 нм, при этом каждая волна может переносить информацию со скоростью до 100 Гбит/с. Скорость передачи зависит от того, какая технология дискретного кодирования данных применяется на каждой волне, например это может быть кодирование технологии SDH или OTN. Говорят, что такая технология образует цифровую оболочку волн DWDM. В настоящее время ведутся работы по повышению скорости передачи информации на одной длине волны до 200 Гбит/с и более.

У технологии DWDM имеется предшественница — технология **волнового мультиплексирования** (Wave Division Multiplexing, WDM), в которой используется существенно меньшее число волн, обычно до восьми, в окнах прозрачности 1310 нм и 1550 нм с разносом несущих в 20 нм (2556 ГГц). Эта технология также называется технологией грубого волнового мультиплексирования (Coarse Wave Division Multiplexing, CWDM), из-за того что волны находятся на большом расстоянии друг от друга, а значит, и сигнал отдельной волны легче выделять из общего светового сигнала (фильтровать) и декодировать.

Мультиплексирование DWDM называется «уплотненным» (dense), из-за того что расстояние между длинами волн в нем существенно меньше, чем в WDM. На сегодня рекомендацией G.694.1 сектора ITU-T для систем DWDM определено четыре *частотных плана* (то есть набора частот, отстоящих друг от друга на некоторую постоянную величину) с шагом (то есть расстоянием между соседними волнами) в 100, 50, 25 и 12,5 ГГц. При этом на практике оборудование DWDM пока работает в основном с первыми двумя частотными планами, что обеспечивает около 40 волн для плана с шагом 100 ГГц и около 80 волн для шага 50 ГГц.

Реализация частотных планов с шагом 50, 25 и 12,5 ГГц предъявляет гораздо более жесткие требования к оборудованию DWDM, особенно в том случае, если каждая волна переносит сигналы со скоростью 10 Гбит/с и выше. Это легко объяснить, если вспомнить, что спектр сигнала тем шире, чем выше частота его модуляции (при фиксированной технике модуляции). Например, спектр сигнала STM-64 шире спектра сигнала STM-16 (рис. 10.8), что приводит к частичному перекрытию сигналов STM-64 у соседних волн при использовании плана 50 ГГц, в то время как при плане 100 ГГц такого перекрытия не происходит.

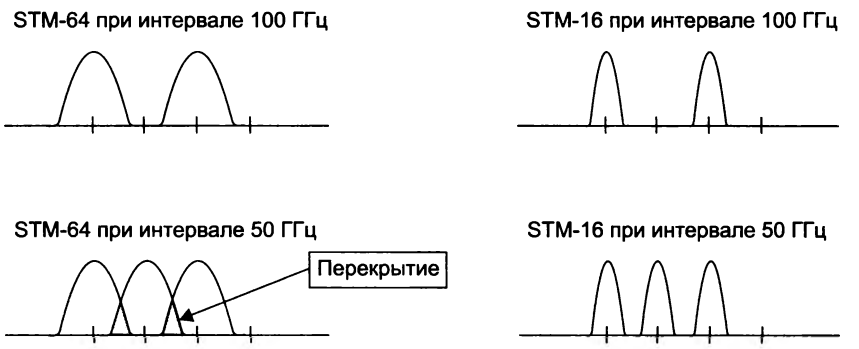


Рис. 10.8. Перекрытие спектра соседних волн для разных частотных планов и скоростей передачи данных

Волоконно-оптические усилители

Практический успех технологии DWDM, оборудование которой уже работает на магистралях многих ведущих мировых операторов связи, во многом определило *появление волоконно-оптических усилителей*. Эти оптические устройства непосредственно усиливают световые сигналы в диапазоне 1550 нм, исключая необходимость промежуточного преобразования их в электрическую форму, как это делают регенераторы, применяемые в сетях SDH. Системы электрической регенерации сигналов весьма дороги, и кроме того, зависят от протокола, так как они должны воспринимать определенный вид кодирования сигнала. Оптические усилители, «прозрачно» передающие информацию, позволяют наращивать скорость магистрали без необходимости модернизировать усилительные блоки.

Наибольшее распространение в волоконно-оптических сетях получили усилители на примесном волокне, то есть волокне, легированном каким-либо редкоземельным элементом. Лазер усилителя, называемый лазером накачки, возбуждает атомы примесей в легированном волокне, при возвращении в нормальное состояние эти атомы излучают свет на той же длине волны и с той же фазой, что и внешний сигнал, требующий усиления. В конце 80-х годов был изобретен усилитель EDFA (Erbium Doped Fiber Amplifier), использующий примеси эрбия. Этот усилитель работает в окне прозрачности 1550 нм, но его полоса усиления имеет ширину 40 нм, в то время как само окно прозрачности имеет ширину около 145 нм (от 1530 до 1675 нм). Поэтому усилитель EDFA обеспечивает усиление только определенной части полного диапазона этого окна, при этом центр полосы усиления может сдвигаться за счет изменения параметров усилителя. На практике усилитель EDFA может работать либо в так называемом C-диапазоне волн окна прозрачности 1550 нм (C-Band, Common Band — обычный диапазон) с границами от 1531 до 1570 нм, либо в L-диапазоне этого окна прозрачности (L-Band, Long band — диапазон длинных волн) с границами от 1570 до 1611 нм, но не в обоих одновременно.

Перспективным типом усилителя для передачи данных на скоростях 100 Гбит/с и выше считается усилитель, который использует эффект рассеяния света Рамана[1]. При использовании рамановского усилителя энергия лазера накачки вызывает распределенное усиление сигнала в самом передающем волокне. Рамановский усилитель обладает более широкой полосой усиления, до 100 нм, что позволяет покрыть как C-, так и L- диапазоны, а значит, передавать большее количество волн в одном волокне. Кроме того, рамановский усилитель вносит меньше нелинейных шумов, а значит, позволяет увеличить максимальную длину участка между оптическими усилителями, которая может достигать 200 км и более.

С успехами DWDM связано еще одно перспективное технологическое направление — **полностью оптические сети**. В таких сетях все операции по мультиплексированию/демультиплексированию, вводу-выводу и кросс-коммутации (маршрутизации) пользовательской информации выполняются без преобразования сигнала из оптической формы в электрическую. Исключение преобразований в электрическую форму позволяет существенно удешевить сеть. Однако возможности оптических технологий пока еще недостаточны для создания масштабных полностью оптических сетей, поэтому их практическое применение ограничено фрагментами, между которыми выполняется электрическая регенерация сигнала.

---

[1] Чандрасекхара Раман открыл эффект рассеяния света в 1928 году, в 1930 он получил за это Нобелевскую премию.

Устройства компенсации дисперсии

Хроматическая дисперсия вносит основной вклад в искажение формы светового сигнала, что, в свою очередь, может приводить к ошибкам в распознавании передаваемых дискретных данных приемниками DWDM. Для уменьшения эффекта хроматической дисперсии в сетях DWDM можно применять специальные волоконно-оптические кабели со смещенной ненулевой дисперсией по стандарту G.655. Этот тип волокна не устраняет дисперсию полностью, но делает ее значительно меньшей, чем при использовании стандартного волокна G.652.

Существуют также **устройства компенсации дисперсии** (Dispersion Compensation Units, **DCU**), которые устанавливаются в промежуточных узлах сети. Такие устройства могут использовать различные способы компенсации разницы в скорости распространения волн различной длины. Очень распространенной техникой является применение в DCU довольно длинного отрезка волокна (компенсационная катушка) со смещенной отрицательной дисперсией, которое компенсирует положительную величину дисперсии стандартного волокна. Компенсационная катушка вносит довольно значительное дополнительное затухание, которое нужно учитывать при проектировании сети DWDM.

Другим распространенным типом DCU является устройство на основе решетки Брега, которое более компактно, чем компенсационная катушка, а кроме того, вносит меньшее дополнительное затухание.

Существуют также устройства компенсации дисперсии поляризации, они существенно сложнее устройств компенсации хроматической дисперсии, их установка требуется только при передаче данных на скоростях выше 10 Гбит/с.

Типовые топологии и узлы сети DWDM

Хронологически первой областью применения технологии DWDM (как и технологии SDH) стало создание сверхдальних высокоскоростных магистралей, имеющих топологию **двухточечной цепи** (рис. 10.9).

Для организации такой магистрали достаточно в ее конечных точках установить **терминальные мультиплексоры DWDM**, а в промежуточных точках — оптические усилители и, возможно, устройства компенсации дисперсии, если этого требует расстояние между конечными точками. Транспондеры (**trans**mitter-re**sponder**), которые обозначены на рис. 10.9 буквой T, преобразуют электрические сигналы, несущие дискретную информацию, поступающую от абонентских устройств пользователей сети DWDM, в оптические сигналы определенной длины волны и обратно.

В приведенной на рисунке схеме дуплексный обмен между абонентами сети происходит за счет однонаправленной передачи всего набора волн по двум волокнам. Существует и другой вариант работы сети DWDM, когда для связи узлов сети используется одно волокно. Дуплексный режим достигается путем двунаправленной передачи оптических сигналов по волокну — половина волн частотного плана передают информацию в одном направлении, половина — в обратном.

Естественным развитием топологии двухточечной цепи является **цепь с промежуточными подключениями**, в которой промежуточные узлы выполняют функции оптических мультиплексоров ввода-вывода DWDM (рис. 10.10).

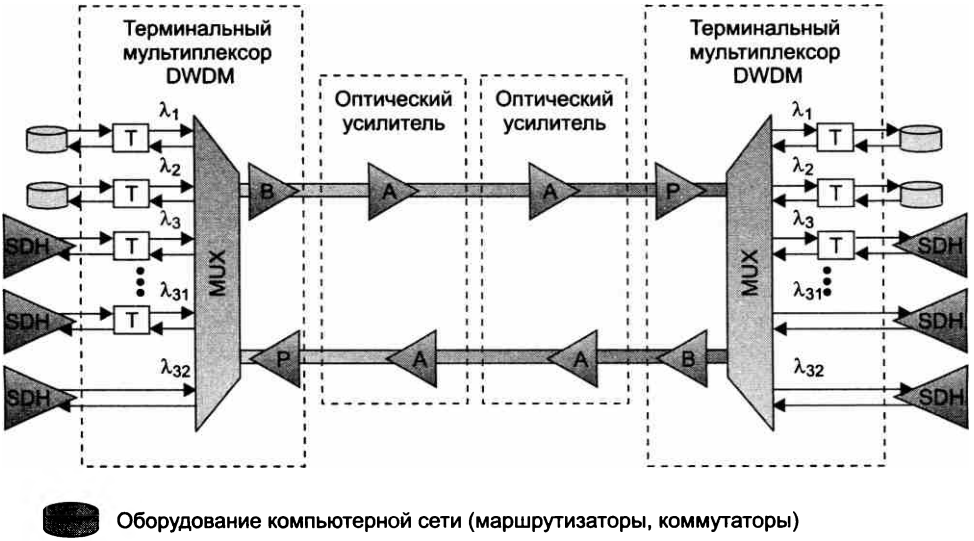


Рис. 10.9. Сверхдальняя двухточечная связь на основе терминальных мультиплексоров DWDM

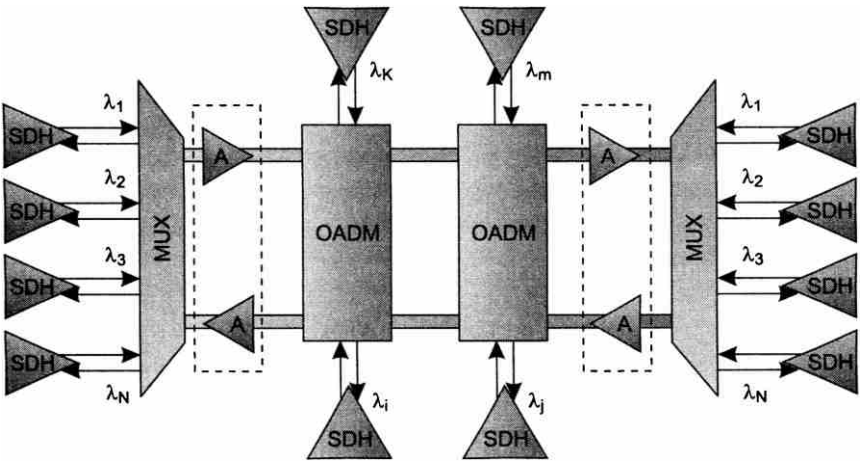


Рис. 10.10. Цепь DWDM с вводом-выводом в промежуточных узлах

Оптические мультиплексоры ввода-вывода (Optical Add-Drop Multiplexer, **OADM**) могут вывести из общего оптического сигнала волну определенной длины и ввести туда сигнал той же длины волны, так что спектр транзитного сигнала не изменится, а соединение будет выполнено с одним из абонентов, подключенных к промежуточному мультиплексору. OADM поддерживает операции ввода-вывода волн сугубо оптическими средствами или с промежуточным преобразованием в электрическую форму.

Кольцевая топология (рис. 10.11) обеспечивает живучесть сети DWDM за счет резервных путей. Методы защиты трафика, применяемые в DWDM, аналогичны методам в SDH (хотя

в DWDM они пока не стандартизованы). Для того чтобы какое-либо соединение было защищено, между его конечными точками устанавливаются два пути: основной и резервный. Мультиплексор конечной точки сравнивает два сигнала и выбирает сигнал лучшего качества (или сигнал, заданный по умолчанию).

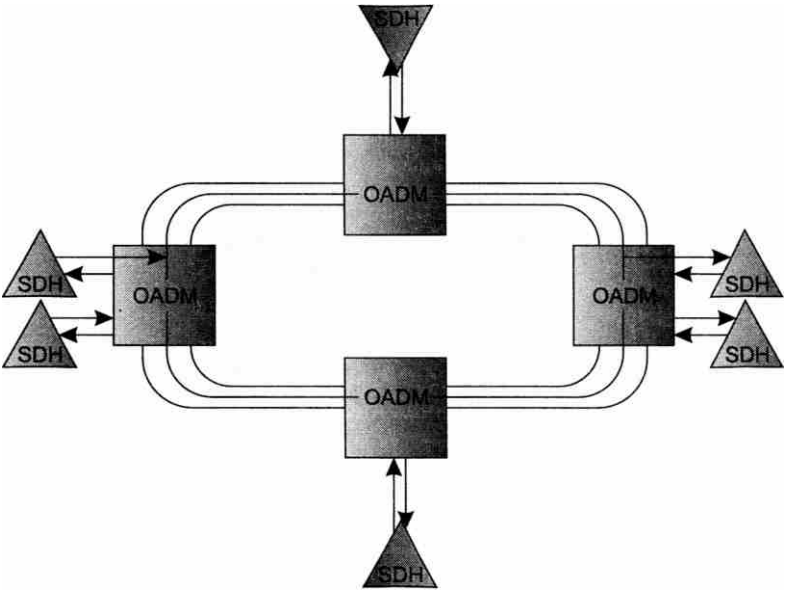


Рис. 10.11. Кольцо мультиплексоров DWDM

По мере развития сетей DWDM в них все чаще будет применяться **ячеистая топология** (рис. 10.12), которая обеспечивает лучшие показатели в плане гибкости, производительности и отказоустойчивости, чем остальные топологии. Однако для реализации ячеистой

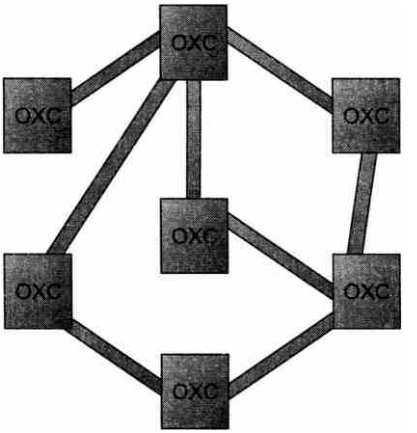


Рис. 10.12. Ячеистая топология сети DWDM

топологии необходимо наличие **оптических кросс-коннекторов** (Optical Cross-Connector, OXC), которые не только добавляют волны в общий транзитный сигнал и выводят их оттуда, как это делают мультиплексоры ввода-вывода, но и поддерживают произвольную коммутацию между оптическими сигналами, передаваемыми волнами разной длины. Возможности оптических кросс-коннекторов по созданию ячеистой топологии оцениваются количеством магистральных связей, которые они могут поддерживать со своими непосредственными соседями по сети. Эти связи проектировщики сетей DWDM называют также *направлениями* (точнее — направлениями маршрутизации). Так, верхний кросс-коннектор на рис. 10.12 поддерживает четыре направления, а нижний — только два. Нетрудно заметить, что обычный мультиплексор ввода-вывода всегда поддерживает только два направления.

Устройство оптических мультиплексоров ввода-вывода

Для выделения волн в мультиплексоре могут использоваться разнообразные оптические механизмы. В оптических мультиплексорах, поддерживающих сравнительно небольшое количество длин волн в волокне, обычно 16 или 32, применяются **тонкопленочные фильтры**. Они состоят из пластин с многослойным покрытием, в качестве такой пластины на практике применяется торец оптического волокна, скошенный под углом 30–45°, с нанесенными на него слоями покрытия. Для систем с большим числом волн требуются другие принципы фильтрации и мультиплексирования.

В мультиплексорах DWDM применяются интегрально выполненные **дифракционные фазовые решетки**, или **дифракционные структуры** (Arrayed Waveguide Grating, AWG). Функции пластин в них выполняют оптические волноводы или волокна. Приходящий мультиплексный сигнал попадает на входной порт (рис. 10.13, *а*). Затем этот сигнал проходит через волновод-пластину и распределяется по множеству волноводов, представляющих дифракционную структуру AWG. Сигнал в каждом из волноводов по-прежнему является мультиплексным, а каждый канал (λ_1, λ_2, ..., λ_N) остается представленным во всех волноводах. Далее происходит отражение сигналов от зеркальной поверхности, и в итоге световые потоки вновь собираются в волноводе-пластине, где происходят их фокусировка и интерференция — образуются пространственно разнесенные интерференционные максимумы интенсивности, соответствующие разным каналам. Геометрия волновода-пластины, в частности расположение выходных полюсов, и значения длины волноводов структуры AWG рассчитываются таким образом, чтобы интерференционные максимумы совпадали с выходными полюсами. Мультиплексирование происходит обратным путем.

Другой способ построения мультиплексора базируется не на одной, а на паре волноводов-пластин (рис. 10.13, *б*). Принцип действия такого устройства аналогичен предыдущему случаю, за исключением того, что здесь для фокусировки и интерференции используется дополнительная пластина.

Интегральные решетки AWG (называемые также **фазарами**) стали одними из ключевых элементов мультиплексоров DWDM. Они обычно применяются для полного демультиплексирования светового сигнала, так как хорошо масштабируются и потенциально могут успешно работать в системах с сотнями спектральных каналов.

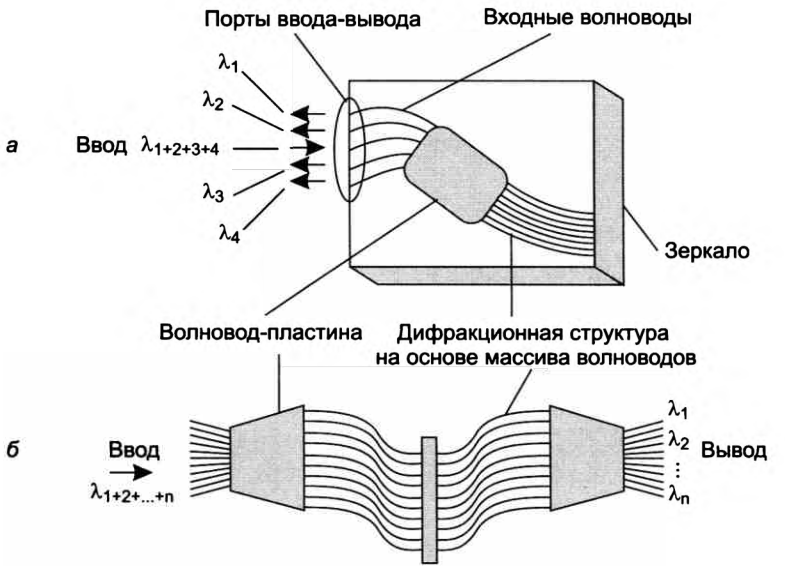


Рис. 10.13. Полное демультиплексирование сигнала с помощью дифракционной фазовой решетки

Устройство оптических кросс-коннекторов

В сетях с ячеистой топологией необходимо обеспечить гибкие возможности для изменения маршрута следования волновых соединений между абонентами сети. Такие возможности предоставляют оптические кросс-коннекторы, позволяющие направить любую из волн входного сигнала каждого порта в любой из выходных портов (конечно, при условии, что никакой другой сигнал этого порта не использует эту волну, иначе необходимо выполнить трансляцию длины волны).

Существуют оптические кросс-коннекторы двух типов:

- **оптоэлектронные кросс-коннекторы** с промежуточным преобразованием в электрическую форму;
- **полностью оптические кросс-коннекторы**, или **фотонные коммутаторы**.

Исторически первыми появились оптоэлектронные кросс-коннекторы, за которыми и закрепилось название оптических кросс-коннекторов. Поэтому производители полностью оптических устройств этого типа стараются использовать для них другие названия: фотонные коммутаторы, маршрутизаторы волн, лямбда-маршрутизаторы. У оптоэлектронных кросс-коннекторов имеется принципиальное ограничение — они хорошо справляются со своими обязанностями при работе на скоростях до 2,5 Гбит/с, но на скоростях 10 Гбит/с и выше габариты таких устройств и потребление энергии превышают допустимые пределы. Фотонные коммутаторы свободны от такого ограничения.

Для коммутации волн в фотонных коммутаторах используются различные оптические механизмы, в том числе дифракционные фазовые решетки AWG и **микроэлектронные механические системы** (Micro-Electro Mechanical System, **MEMS**).

MEMS представляет собой набор подвижных зеркал очень маленького (диаметром менее миллиметра) размера (рис. 10.14). Коммутатор на основе MEMS включается в работу после демультиплексора, когда исходный сигнал уже разделен на составляющие волны. За счет поворота микрозеркала на заданный угол исходный луч определенной волны направляется в соответствующее выходное волокно. Затем все лучи мультиплексируются в общий выходной сигнал.

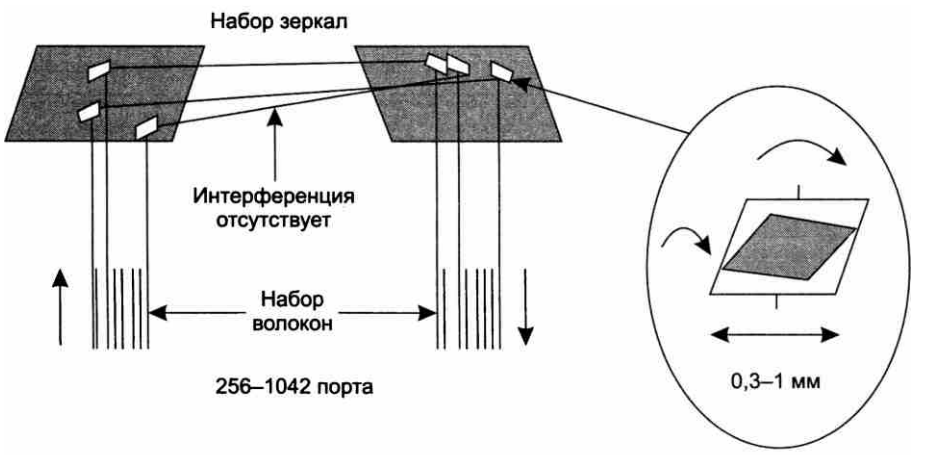


Рис. 10.14. Микроэлектронная механическая система кросс-коммутации

Для коммутации волн могут также использоваться устройства на жидких кристаллах.

Реконфигурируемые оптические мультиплексоры ввода-вывода (Reconfigurable Optical Add-Drop Multiplexors, **ROADM)** представляют собой *новое поколение фотонных кросс-коннекторов*, позволяющих удаленно динамически изменять маршрутизацию различных волн, передаваемых мультиплексором.

До появления ROADM добавление новой волны (операция Add) и выведение ее из общего сигнала (операция Drop) обычно требовали физической установки нового модуля на шасси мультиплексора и его локального конфигурирования, что, естественно, требовало посещения инженером точки присутствия оператора, в которой был установлен мультиплексор. Ранние сети DWDM были достаточно статическими в отношении реконфигурации вводимых и выводимых потоков данных, поэтому с необходимостью выполнять эту операцию путем физической перекоммутации операторы мирились. Развитие сетей DWDM привело к усложнению их топологии и повышению динамизма, когда появление новых клиентов сети стало достаточно частым явлением, а значит, операции добавления или выведения волн из магистрали стали выполнятся регулярно и требовать более эффективной поддержки.

Типичная схема ROADM показана на рис. 10.15. Мультиплексор на этом рисунке поддерживает три магистральных направления (порты маршрутов 1, 2 и 3 связывают его с другими мультиплексорами), а также три банка транспондеров, связанных через клиентский кросс-коннектор с портами ввода-вывода.

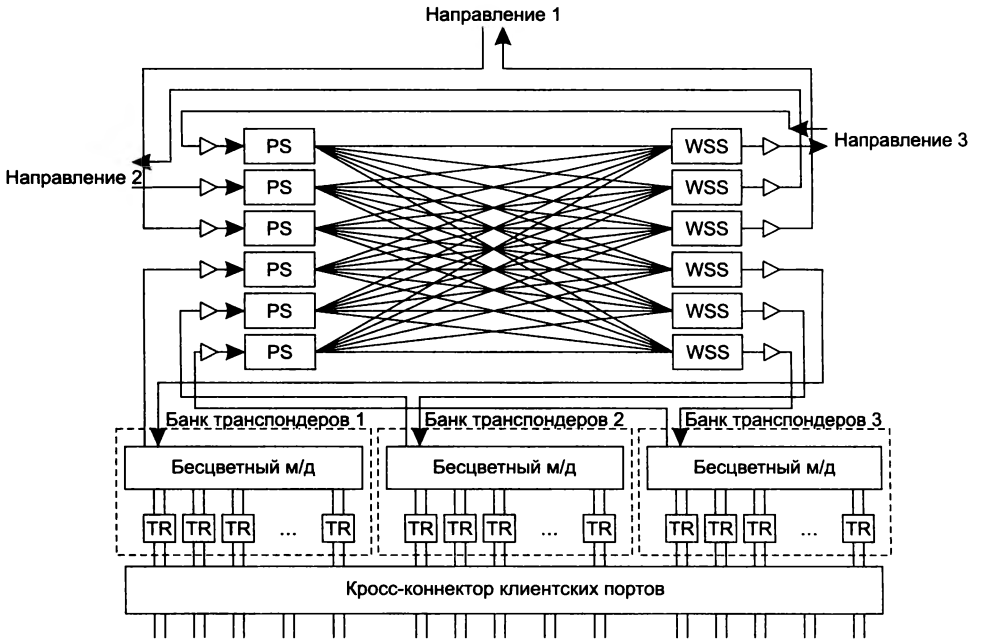


Рис. 10.15. Структурная схема ROADM

Коммутационная матрица ROADM состоит из **устройств селективной коммутации волн** (Wavelength Selective Switch, **WSS**). Каждое устройство WSS связано с одним из выходных магистральных портов ROADM или с одним из банков транспондеров. На каждое из устройств WSS с помощью оптического *разветвителя мощности* (Power Splitter, PS) подается исходный «цветной» сигнал (то есть сигнал, содержащий сигналы всех волн DWDM) с входного магистрального порта или от банка транспондеров. Основной функцией WSS является расщепление поступающих на его входы цветных сигналов на составляющие волны, выбор определенных волн из каждого сигнала и их направление (маршрутизирование) в свой выходной порт. Коммутатор WSS может быть реализован на основе систем MEMS, жидких кристаллов или любого другого механизма, выполняющего фотонную коммутацию.

Количество транспондеров в банках соответствует количеству волн, поддерживаемых сетью DWDM, например 80. Клиентские порты ввода-вывода волн соединены с помощью отдельного кросс-коннекта с банками транспондеров таким образом, что любой порт может быть соединен с любым транспондером любого банка. Такая гибкая коммутация позволяет разделять дорогостоящие транспондеры между клиентскими портами и таким образом повысить коэффициент их использования.

Каждый банк транспондеров соединен со своим разветвителем мощности с помощью мультиплексора, который объединяет сигналы отдельных волн, генерируемых транспондерами, в общий «цветной» сигнал. Соответственно «цветной» сигнал от коммутатора WSS, выделенного для обслуживания определенного банка транспондеров, расщепляется на сигналы

отдельных волн с помощью демультиплексора банка, передающего эти сигналы на соответствующие транспондеры. Количество банков транспондеров должно соответствовать количеству N направлений, поддерживаемых ROADM, так как одна и та же волна может быть выведена из любого направления, следовательно, может быть представлена N раз в наборе пользовательских портов ввода-вывода (в нашем примере $N = 3$).

Особенностью представленной архитектуры ROADM является возможность вывести любую волну из любого направления в любой пользовательский порт ввода-вывода. Говорят, что ROADM, обладающий такой функциональностью, является **бесцветным** (colorless) **ненаправленным** (directionless). Необходимо подчеркнуть, что не каждое устройство ROADM является бесцветным ненаправленным, так как название устройства этого типа отражает только возможность его программного реконфигурирования, но не его степень. Например, устройства ROADM первого поколения имели только один банк транспондеров, так что пользовательские порты ввода-вывода были физически привязаны к определенной волне и программное реконфигурирование позволяло вывести определенную волну только в определенный порт, изменение волны для некоторого пользователя требовало физического переключения его кабеля к другому порту. Поэтому такие устройства ROADM не могли быть названы бесцветными. Аналогично обстояло дело и с ненаправленностью, так как эти устройства ROADM были рассчитаны на кольцевые или линейные топологии и поддерживали только одно направление.

Новое поколение устройств ROADM способно поддерживать еще одно полезное свойство — они могут быть **неблокирующими** (contentionless). Неблокирующее устройство ROADM позволяет обслуживать любой запрос на маршрутизацию волны независимо от того, какие запросы на маршрутизацию других волн он уже обслуживает.

Сети OTN

Причины и цели создания

Сети DWDM не являются собственно цифровыми сетями, так как они лишь предоставляют пользователям отдельные спектральные каналы, являющиеся не более чем несущей средой. Для того чтобы передавать по такому каналу цифровые данные, необходимо каким-то образом договориться о методе модуляции или кодирования двоичных данных, а также предусмотреть такие важные механизмы, как контроль корректности данных, исправление битовых ошибок, обеспечение отказоустойчивости, оповещение пользователя о состоянии соединения и т. п.

Исторически мультиплексоры DWDM были также и мультиплексорами SDH, то есть в каждом из волновых каналов для решения перечисленных задач они использовали технику SDH. Однако по прошествии некоторого времени эксплуатации сетей SDH/DWDM стали заметны определенные недостатки, связанные с применением технологии SDH в качестве основной технологии передачи цифровых данных по спектральным каналам DWDM.

Перечислим эти недостатки.

- *Недостаточная эффективность кодов FEC, принятых в качестве стандарта SDH.* Это препятствует дальнейшему повышению плотности спектральных каналов в мульти-

плексорах DWDM. Логика здесь следующая: при увеличении количества спектральных каналов в оптическом волокне увеличивается взаимное влияние их сигналов, следовательно, возрастают искажения сигналов и, как следствие, битовые ошибки при передаче цифровых данных по этим спектральным каналам. Если же процедуры FEC настолько эффективны, что позволяют «на лету» устранить значительную часть этих ошибок, то этими ошибками можно пренебречь и увеличить количество спектральных каналов. Или же можно не увеличивать количество каналов, а увеличить длину нерегенерируемых секций сети.

- *Слишком «мелкие» единицы коммутации для магистральных сетей, работающих на скоростях 10, 40 и 100 Гбит/с.* Учет таких клиентских каналов, как каналы со скоростью 1,5, 2 или 34 Мбит/с, усложняет оборудование сети, поэтому желательно наличие единиц коммутации, более соответствующих битовой скорости современного клиентского оборудования. Механизм виртуальной конкатенации SDH частично решает эту проблему, но в целом она остается.
- *Не учтены особенности трафика различного типа.* Разработчиками технологии SDH принимался во внимание только голосовой трафик, тогда как сегодня преобладающим является компьютерный трафик.

На преодоление этих недостатков нацелена новая технология **оптических транспортных сетей** (Optical Transport Network, OTN), которая обеспечивает передачу и мультиплексирование цифровых данных по волновым каналам DWDM более эффективно, чем SDH. В то же время сети OTN обеспечивают обратную совместимость с SDH, так как для мультиплексоров OTN трафик SDH является одним из видов пользовательского трафика наряду с такими клиентами, как Ethernet и GFP.

Нужно отметить, что технология OTN не заменяет технологии DWDM, а дополняет ее волновые каналы «цифровой оболочкой»[1] — этим термином называют кадры данных OTN, позволяющие передавать в канале DWDM дискретные данные пользователей.

Архитектура сетей OTN описана в стандарте ITU-T G.872, а наиболее важные технические аспекты работы узла сети OTN описаны в стандарте G.709.

Иерархия скоростей

Технология OTN многое взяла от SDH, в том числе коэффициент кратности скоростей 4 для построения своей иерархии скоростей. Однако начальная скорость иерархии скоростей OTN гораздо выше, чем у SDH: 2,5 Гбит/с вместо 155 Мбит/с.

В настоящее время стандартизована четырехступенчатая иерархия скоростей OTN, которые выбраны так, чтобы прозрачным образом передавать клиентские кадры вместе со служебными заголовками (табл. 10.3).

Из-за наличия служебной информации в заголовках кадров OTN приведенные значения скоростей кадров OTN выше скоростей клиентских данных, вложенных в эти кадры.

[1] Термин «цифровая оболочка» (digital wrapper) иногда даже используется в качестве названия самой технологии OTN.

Таблица 10.3. Иерархия скоростей технологии OTN

Интерфейс G.709	Битовая скорость кадров OTN (Гбит/с)	Клиентский кадр	Битовая скорость клиента (Гбит/с)
OTU1	2,666	STM-16	2,488
OTU2	10,709	STM-64	9,953
OTU3	43,018	STM-256	39,813
OTU4	111,8	100G Ethernet	100

Стек протоколов OTN

Стек протоколов OTN состоит из четырех уровней.

На рис. 10.16 показана обобщенная архитектура сети OTN и области применения протокола каждого уровня, а на рис. 10.17 — иерархия протоколов OTN.

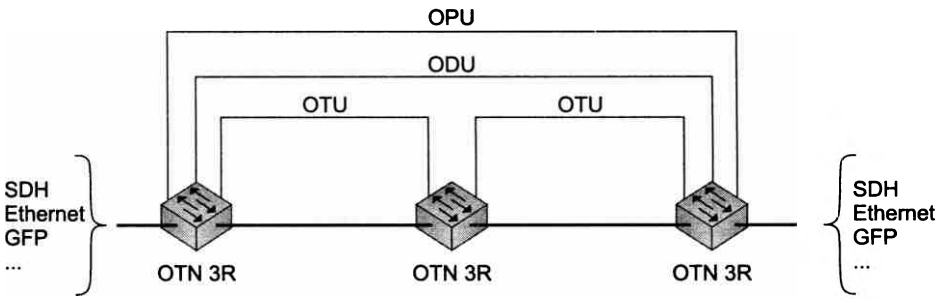


Рис. 10.16. Сеть OTN и распределение протоколов

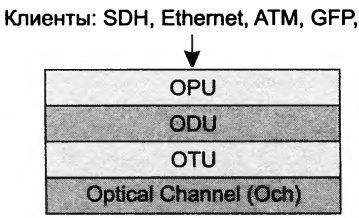


Рис. 10.17. Иерархия протоколов OTN

Нижний уровень протоколов составляет **оптический канал** (Optical Channel, Och); обычно это спектральный канал DWDM. Данный уровень примерно соответствует фотонному уровню технологии SDH.

Протокол OPU (Optical Channel Payload Unit — блок пользовательских данных оптического канала) ответствен за доставку данных между пользователями сети. Он обеспечивает инкапсуляцию пользовательских данных, таких как кадры SDH или Ethernet, в блоки OPU, выравнивание скоростей передачи пользовательских данных и блоков OPU, а на приемной стороне извлекает пользовательские данные и передает их пользователю. В зависимости от скорости передачи данных этому протоколу соответствуют блоки OPU1, OPU2, OPU3

и OPU4. Для выполнения своих функций протокол OPU добавляет к пользовательским данным свой заголовок OPU OH (OverHead). Блоки OPU не модифицируются сетью.

Протокол ODU (Optical Channel Data Unit — блок данных оптического канала), так же как и протокол OPU, работает между конечными узлами сети OTN. В его функции входят мультиплексирование и демультиплексирование блоков OPU, то есть, например, мультиплексирование четырех блоков OPU1 в один блок OPU2. Кроме того, протокол ODU поддерживает функции мониторинга качества соединений в сети OTN. Это протокол формирует блоки ODU требуемой скорости, добавляя к соответствующим блокам OPU свой заголовок. Протокол ODU является аналогом протокола линии SDH.

Протокол OTU (Optical Channel Transport Unit — транспортный блок оптического канала) работает между двумя соседними узлами сети OTN, которые поддерживают функции электрической регенерации оптического сигнала, называемые также функциями 3R (retiming, reshaping, and regeneration). Основное назначение этого протокола — контроль и исправление ошибок с помощью кодов FEC. Этот протокол добавляет к блоку ODUk свой концевик, содержащий код FEC, образуя блок OTUk. Протокол OTU соответствует протоколу секции SDH. Блоки OTUk помещаются непосредственно в оптический канал.

Кадр OTN

Кадр OTN обычно представляют в виде матрицы, состоящей из 4080 столбцов-байтов и четырех строк (рис. 10.18).

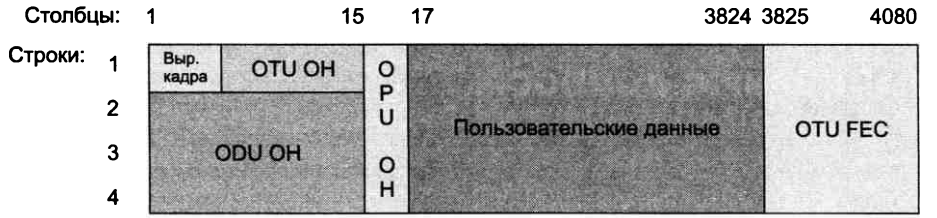


Рис. 10.18. Формат кадра OTN

Кадр состоит из поля пользовательских данных (Payload) и служебных полей блоков OPU, ODU и OTU. Формат кадра не зависит от уровня скорости OTN, то есть он, например, одинаков для блоков OPU1/ODU1/OTU1 и OPU2/ODU2/OTU2.

Поле пользовательских данных располагается с 17-го по 3824-й столбец и занимает все четыре строки кадра, а заголовок блока OPU занимает столбцы 15 и 16 также в четырех строках. При необходимости заголовок OPU OH может занимать несколько кадров подряд (в этих случаях говорят о мультикадре OTN), например такой вариант встречается в том случае, когда нужно описать структуру поля пользовательских данных, мультиплексирующую несколько блоков OPU более низкого уровня.

Блок ODU представлен только заголовком ODU OH (формально он также имеет поле данных, в которое помещен блок OPU), а блок OTU состоит из заголовка OTU OH и концевика OTU FEC, содержащего код коррекции ошибок FEC.

Начинается кадр с небольшого поля выравнивания кадра, необходимого для распознавания начала кадра.

Выравнивание скоростей

Как и в других технологиях, основанных на синхронном мультиплексировании TDM, в технологии OTN решается проблема выравнивания скоростей пользовательских потоков со скоростью передачи данных мультиплексора.

Работа механизма выравнивания OTN зависит от того, какой режим отображения нагрузки на кадры OTM поддерживается для данного пользовательского потока — синхронный или асинхронный. В режиме **синхронного отображения нагрузки** (Synchronous Bit Mapping, SBM) мультиплексор OTM синхронизирует прием и передачу данных от синхроимпульсов, находящихся в принимаемом потоке пользовательских данных. Этот режим рассчитан на пользовательские протоколы, данные которых хорошо синхронизированы и содержат в заголовке специальные биты синхронизации (такие, как SDH). В этом случае механизм выравнивания фактически простаивает, так как скорость передачи данных всегда равна скорости их поступления, поэтому выравнивать нечего.

В режиме **асинхронного отображения нагрузки** (Asynchronous Bit Mapping, ABM) мультиплексор OTN синхронизируется от собственного источника синхроимпульсов, который не зависит от пользовательских данных (это может быть любой из способов синхронизации, рассмотренных в разделе, посвященном технологии PDH). В этом случае рассогласование скоростей неизбежно, и поэтому задействуется механизм выравнивания.

Для выравнивания скоростей в кадре OTN используются два байта: байт возможности положительного выравнивания (Positive Justification Opportunity, PJO) и байт возможности отрицательного выравнивания (Negative Justification Opportunity, NJO). Байт PJO находится в поле пользовательских данных, а байт NJO — в заголовке OPU OH. В тех случаях, когда при помещении пользовательских данных скорость выравнивать не нужно, мультиплексор помещает все байты пользовательских данных в байты поля данных, применяя в том числе байт PJO. В тех случаях, когда скорость пользовательского потока меньше скорости мультиплексора и ему не хватает байта для заполнения поля данных, то в байт PJO вставляется «заполнитель», который представляет собой байт с нулевым значением — так выполняется положительное выравнивание. А если скорость пользовательского потока больше скорости мультиплексора, лишний байт пользовательских данных помещается в поле NJO — так происходит отрицательное выравнивание.

Для того чтобы конечный мультиплексор сети правильно выполнил демультиплексирование пользовательских данных, ему нужна информация о том, каким образом в кадре использованы байты NJO и PJO. Такая информация хранится в поле управления выравниванием (Justification Control, JC), два бита которого показывают, какое значение помещено в каждый из байтов NJO и PJO.

Указатель на начало пользовательских данных в технологии OTN отсутствует. Таким образом, вставка байта делает механизм выравнивания OTN похожим на PDH, где имеет место вставка битов и соответствующие признаки такой вставки (отрицательное выравнивание).

Мультиплексирование блоков

При мультиплексировании блоков ODU поле пользовательских данных блока OPUk разбивается на так называемые **трибутарные слоты** (Tributary Slot, TS), в которые помещаются данные блока OPUk-1.

На рис. 10.19 показан пример мультиплексирования четырех блоков ODU1 в один блок ODU2. Как видно из рисунка, поле данных блока OPU2 разбито на трибутарные слоты TribSlot1, TribSlot2, TribSlot3 и TribSlot4, последовательность которых повторяется. Каждый из этих четырех трибутарных слотов предназначен для переноса части поля данных одного из блоков OPU1. Здесь используется техника чередования данных скорости более низкого уровня иерархии скоростей в поле данных блока более высокой скорости иерархии скоростей, которая типична для технологий синхронного временно́го мультиплексирования. Эта техника обеспечивает выполнение операций мультиплексирования и демультиплексирования «на лету» (без промежуточной буферизации), так как частота появления порций данных OPU1 в блоке ODU2 соответствует частоте их появления в том случае, как если бы они передавались на скорости OPU1.

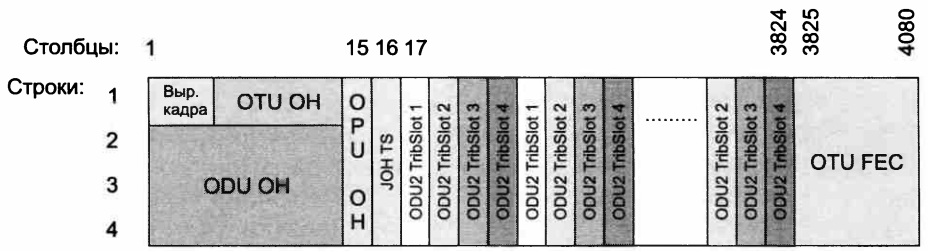


Рис. 10.19. Мультиплексирование блоков ODU1 в блок ODU2

Техника мультиплексирования блоков ODU1 и ODU2 в блок ODU3 аналогична, если не считать того, что в блоке OPU3 используется 16 различных трибутарных слотов, что позволяет поместить в него 16 блоков ODU1 или 4 блока ODU2 (в этом случае одной порции OPU2 соответствует четыре трибутарных слота ODU3). Существуют два типа трибутарных слотов: первый тип соответствует скорости данных 2,5 Гбит/с, второй — 1,25 Гбит/с (он был введен для более эффективного мультиплексирования данных 1G Ethernet).

Информация об использовании трибутарных слотов хранится в специальном разделе поля OPU2 OH или OPU3 OH. Этот раздел может также запоминать информацию о *виртуальной конкатенации блоков ODU1 или ODU2* — эта техника также поддерживается в сетях OTN.

Гибкое мультиплексирование

Технология OTN совершенствуется, позволяя все более эффективно и гибко упаковывать пользовательские данные различных протоколов в кадры OPU/ODU/OTU. Первоначально (в стандартах 2001 года) было определено всего три уровня иерархии скоростей и соответственно три формата кадров, ODU1, ODU2 и ODU3, которые были выбраны для передачи данных STM-16, STM-64 и STM-256. Затем эти типы кадров были дополнены форматами ODU2e для передачи кадров 10G Ethernet (так как для их передачи требуется немного большая скорость, чем для передачи кадров STM-64). Для более эффективного мультиплексирования данных 1G Ethernet был также определен формат ODU0 с битовой скоростью 1,25 Гбит/с (при передаче через сеть OTN два таких кадра всегда упаковываются в кадр ODU1). Появление стандарта 100G Ethernet вызвало стандартизацию нового уровня скорости OTN с форматом кадра ODU4.

Появление новых уровней скоростей привело к увеличению различных вариантов мультиплексирования кадров более низких уровней в кадры более высоких уровней, что создавало проблему стандартизации этих вариантов при введении каждого нового формата кадров ODUk, при этом число вариантов мультиплексирования росло нелинейно с возрастанием количества типов кадров. Поэтому разработчики стандартов OTN нашли более фундаментальное решение этой проблемы, введя формат кадра ODUflex с произвольной битовой скоростью и процедуру обобщенного отображения нагрузки (Generic Bit Mapping, GMP).

Формат ODUflex определен в двух вариантах.

Первый вариант ODUflex(CBR) используется для упаковки синхронных данных, таких как данные оцифрованного голоса или видео, имеющего постоянную битовую скорость (Constant Bit Rate, CBR). При помещении пользовательских данных в кадр ODUflex применяется синхронная (SBM) или асинхронная (ABM) процедура отображения нагрузки в зависимости от того, как синхронизируется приемник, от сигналов пользователя или локально. Скорость пользовательского потока данных может быть произвольной, не совпадающей ни с одним уровнем иерархии скоростей OTN.

Второй вариант ODUflex(GFP) предназначен для мультиплексирования данных компьютерных протоколов, таких как Ethernet или MPLS. Пакет данных полностью буферизуется приемником, а затем с помощью процедуры GFP помещается в поле данных кадра ODUflex(GFP).

После того как кадр ODUflex сформирован — это относится как к кадру ODUflex(CBR), так и к кадру ODUflex(GFP), — он помещается в трибутарные слоты кадров ODUk более высокого уровня с помощью процедуры обобщенного отображения нагрузки. При этом используются трибутарные слоты 1,25 Гбит/с. Например, кадр ODUflex(CBR) со скоростью 3 Гбит/с может быть помещен в три трибутарных слота кадра — ODU2, ODU3 или ODU4. Так как суммарная скорость трех трибутарных слотов превышает скорость данных кадра ODUflex(CBR), то определенная часть байтов этих слотов содержит байты-заполнители.

С помощью процедуры BMP можно вычислить позиции байтов-заполнителей в кадрах ODUk и таким образом правильно демультиплексировать данные кадров ODUflex при приеме. Для этого в процедуре GMP используется алгоритм «Сигма/Дельта», в котором делается попытка разместить данные в поле избыточного размера наиболее равномерным способом. Например, в том случае, когда поле данных кадра ODUk, в которое помещаются данные ODUflex, оказывается в два раза больше, чем нужно, байты ODUflex помещаются в каждый четный байт поля ODUk, а в нечетные байты помещаются заполнители.

Алгоритм «Сигма/Дельта» использует в своих вычислениях два параметра:

- Pserver — количество слов данных, помещающихся в поле данных кадра ODUk (этот кадр является сервером для данных кадра ODUflex);
- Cn(t) — количество слов данных, которое нужно поместить в поле данных текущего кадра ODUk.

Данные из кадра ODUflex помещаются в кадр ODUk словами, размер которых кратен количеству трибутарных слотов, используемых в кадре ODUk для передачи данных ODUflex: при наличии одного трибутарного слота он равен 8 бит, то есть 1 байт соответственно для N слотов он равен $N \times 8$ бит, или N байт.

Значение Pserver фиксировано и зависит от количества трибутарных слотов, необходимых для передачи данных кадров ODUflex. Значение Cn(t) зависит от разницы в скорости данных ODUflex и ODUk, оно меняется от кадра к кадру как из-за пульсирующего характера компьютерного трафика в случае кадров ODUflex(GFP), так и из-за необходимости помещать в кадр ODUk целое число слов — последнее справедливо как для случая ODUflex(CBR), так и для случая ODUflex(GFP). Поэтому при мультиплексировании данных передатчик помещает текущее значение Cn(t) в заголовок каждого кадра ODUk, так что приемник может правильно демультиплексировать данные.

Алгоритм «Сигма/Дельта» может быть описан с помощью накапливающего сумматора. При обработке каждого нового кадра ODUk сумматор обнуляется. Затем для каждой новой позиции слова *j* к содержимому сумматора добавляется значение Cn(t) и результат сравнивается с Pserver. Если результат оказывается меньше, чем Pserver, то в позицию *j* помещается заполнитель, а содержимое сумматора не изменяется. Если же результат оказывается больше или равен Pserver, то в позицию *j* помещается слово данных, а из содержимого сумматора вычитается Pserver.

Пример заполнения поля данных ODUk показан на рис. 10.20.

Рис. 10.20. Пример работы алгоритма «Сигма/Дельта»

Для иллюстрации работы алгоритма выбран условный кадр, состоящий из одного трибутарного слота (это определяет размер слова в один байт) с размером поля данных в 10 байт (Pserver = 10). Пусть в этот кадр нужно поместить три слова данных (Cn(t) = 3).

На рисунке показаны изменения содержимого накапливающего сумматора *S* при последовательной оценке каждой из 10 позиций поля данных кадра. Значение сумматора три раза превышает или становится равным размеру поля данных — для позиций 4, 7 и 10, поэтому в эти позиции помещаются слова данных, а в остальные позиции — слова-заполнители. При отправке данного кадра в его заголовок помещается значение Cn(t) = 3, так что приемник может найти позиции, в которых находятся данные, применив тот же алгоритм «Сигма-Дельта».

Техника GMP используется не только при помещении данных ODUflex в трибутарные блоки кадров ODUk, но и в некоторых вариантах мультиплексирования поля данных кадра ODU в трибутарные слоты кадра ODU более высокого уровня, например ODU2e в ODU3, ODU2 или ODU3 в ODU4.

Коррекция ошибок

В OTN применяется процедура прямой коррекции ошибок (FEC), в которой используются коды Рида—Соломона RS(255, 239). В этом самокорректирующемся коде данные кодируются блоками по 255 байт, из которых 239 байт являются пользовательскими, а 16 байт представляют собой корректирующий код. Коды Рида—Соломона позволяют исправлять до 8 ошибочных байтов в блоке из 255 байт, что является очень хорошей характеристикой для самокорректирующего кода.

Применение кода Рида—Соломона позволяет улучшить отношение мощности сигнала к мощности шума на 5 дБ при уменьшении уровня битовых ошибок с 10^{-3} (без применения FEC) до 10^{-12} (после применения FEC). Этот эффект дает возможность увеличить расстояние между регенераторами сети на 20 км или использовать менее мощные передатчики сигнала.

Передача данных на скорости 100 Гбит/с

Постоянный рост объема трафика Интернета заставляет разработчиков технологий передачи данных осваивать все новые уровни иерархии скоростей. Ранние версии технологии OTN предусматривали потолок скорости в 10 Гбит/с, на эту скорость отдельного оптического канала также ориентировались разработчики технологии и оборудования DWDM. Однако уже к середине первой декады 2000-х годов этой скорости стало не хватать для качественного обслуживания растущего трафика, поэтому начались работы по созданию оборудования DWDM и OTN, способного работать с оптическими каналами на скоростях 100 Гбит/с. Иерархия скоростей OTN с коэффициентом кратности 4 диктовала введение скоростей 40 и 160 Гбит/с, однако работы над стандартом 100G Ethernet заставили нарушить стройность этой иерархии и сосредоточить усилия на поддержке скорости 100 Гбит/с. Скорость 40 Гбит/с оказалась промежуточным рубежом, на котором опробовались новые идеи и методы для достижения скорости 100 Гбит/с, а их потребовалось немало. Рассмотрим основные из них.

Новые форматы модуляции сигнала

В передатчиках оптического сигнала со скоростью не выше 10 Гбит/с применяется очень простой вид модуляции — кодирование интенсивностью света IDMM, то есть использование периодов передачи светового сигнала полной интенсивности и периодов темноты. Однако применение этого вида модуляции для более высоких скоростей сталкивается со сложностями.

Так, простое 10-кратное наращивание тактовой частоты передатчика с модуляцией IDMM приводит к необходимости генерации символов кода с частотой около 100 Гбод (точнее — 112 Гбод из-за необходимости передачи как пользовательских данных со скоростью 100 Гбит/с, так и служебной информации заголовков кадров OTN), а это предъявляет очень высокие требования к быстродействию электронных схем передатчика. При этом ширина спектра такого сигнала требует сетки частот DWDM с шагом 200 ГГц, то есть сокращает максимально количество волн магистрали DWDM до 20.

Прямое увеличение тактовой частоты передатчика приводит также к такому негативному эффекту, как увеличение отношения сигнала к шуму, а значит, и к снижению реальной пропускной способности канала (вспомните формулу Шеннона). Кроме того, при этом хроматическая дисперсия возрастает в 100 раз, а поляризационная — в 10.

Поэтому разработчики попытались решить проблему за счет применения более сложных, чем IDMM, форматов модуляции, использующих более двух кодовых символов.

Код PM-QPSK (Polarization Multiplexing — поляризационное мультиплексирование, Quadrature Phase Shift Keying — квадратурная фазовая манипуляция) стал основным кодом для передачи данных в оптических сетях со скоростью 100 Гбит/с. Этот код использует две моды поляризации света, при этом в каждой из мод данные кодируются с помощью четырех значений фазы сигнала. В результате сигнал кода имеет 16 различных состояний, так как две моды поляризации образуют два независимых канала, сигнал каждого из которых имеет 4 состояния ($4 \times 4 = 16$). В каждом такте кода PM-QPSK передается 4 бита дискретной информации. Ширина спектра такого сигнала позволяет передавать его в сетке 50 ГГц.

Теоретическое значение частоты модуляции символов кода при использовании 4 бит на символ должно составлять 25 Ггц, однако из-за накладных расходов в виде заголовков кадра OUT4 и поля FEC частота модуляции символов PM-QPSK меняется в пределах 28-32 Гбод, такую частоту гораздо легче реализовать в электронных схемах управления модуляцией передатчика, чем 100 Гбод, необходимых для модуляции IMDD.

Когерентное распознавание кодов и цифровые сигнальные процессоры

Распознавание кодов, переносимых сигналами PM-QPSK, имеющими 16 состояний, представляет собой сложную задачу для приемника оптического узла. Для ее решения была привлечена техника когерентного распознавания, которая внесла существенный вклад в становление технологии 100G.

Когерентное распознавание (coherent detection) основано на смешении поступающего сигнала с сигналом локального источника, длина волны которого совпадает с центром спектра принимаемого сигнала. Узкополосный фильтр на входе приемника отрезает все составляющие входного сигнала, выходящие за границы диапазона данного спектрального канала. На выходе смесителя образуется электрический сигнал промежуточной частоты с амплитудой и фазой, которые пропорциональны амплитуде и фазе входного сигнала. Анализ электрического сигнала позволяет более надежно распознать двоичную информацию, содержащуюся в исходном оптическом сигнале, и тем самым повысить чувствительность приемника в условиях зашумленного сигнала, которым является сигнал с 16 состояниями.

Когерентное распознавание было предметом интенсивных исследований в 1970-е и 1980-е годы, но изобретение оптического усилителя обеспечило гораздо более простой и дешевый метод решения проблемы чувствительности приемника за счет усиления сигнала и интерес к этому способу детектирования угас.

Сегодня технология когерентного распознавания снова является предметом научных исследований и коммерческих разработок, так как ее потенциал был раскрыт за счет обработки оцифрованного сигнала **цифровым сигнальным процессором** (Digital Signal Processor,

DSP). Применение мощных математических методов анализа оцифрованных сигналов позволило не только улучшить чувствительность когерентного приемника (терпимость к шуму увеличивается на 2–4 дБ), но и компенсировать на приемной стороне линейные искажения оптического сигнала — хроматическую и поляризационную дисперсию. В результате *оптические сети, построенные на основе приемников с когерентным распознаванием, не нуждаются в установке в промежуточных точках сети устройств компенсации хроматической и поляризационной дисперсий.*

FEC

При передаче данных со скоростью 100 Гбит/с аппаратная реализация алгоритмов коррекции ошибок на основе кодов Рида—Соломона в передатчиках и приемниках DWDM/OTN стала применяться повсеместно. Коррекция ошибок позволяет снизить требования к уровню отношения сигнала к шуму, а значит, применять менее мощные передатчики и работать на более протяженных отрезках волокна.

Скорость 100 Гбит/с для сетей OTN была стандартизована ITU-T в 2012 году, и сегодня большинство сетей OTN/DWDN операторов связи уже внедрило 100-гигабитное оборудование на магистралях своих сетей.

На пути к терабитным скоростям

Как это постоянно происходит с телекоммуникационной отраслью, начало внедрения сетей нового поколения для операторов сетей одновременно является началом работ над технологиями следующего поколения для исследовательских центров университетов и производителей телекоммуникационного оборудования. Это обстоятельство оказалось справедливым и для технологий DWDM и OTN — одновременно с внедрением магистральных сетей 100 Гбит/с начались работы по разработке оборудования, способного передавать данные с более высокими скоростями — 200 и 400 Гбит/с, 1 Тбит/с. Начальные этапы таких работ (на начало 2015 года) выявили основные направления, на которых сосредоточились разработчики технологии «свыше 100 Гбит/с» и которые мы кратко рассмотрим в этом разделе.

Усовершенствованные форматы модуляции

Замена простой модуляции IMDD с двумя состояниями сигнала на более сложный и эффективный формат PM-QPSK с 16 состояниями в сочетании с когерентным распознаванием обеспечила успех технологии 100 Гбит/с. Разработчики технологии «свыше 100 Гбит/с» также собираются использовать когерентное распознавание в своем оборудовании, однако сохранение модуляции PM-QPSK не сможет обеспечить необходимый скачок скорости. Объясняется это прежде всего возможностями современных электронных схем управления модуляцией в передатчиках; так, например, для работы на скорости 1 Тбит/с передатчику нужно модулировать оптический сигнал PM-QPSK с частотой около 270–320 Гбод вместо 27–32 Гбод, на которой работает электроника передатчика 100 Гбит/с. В настоящее время электроника такого уровня пока не достигла, и по некоторым оценкам, понадобится около 10 лет, чтобы его достичь.

Естественным путем преодоления ограничения в быстродействии электронных схем является дальнейшее повышение количества состояний модулированного сигнала.

Сегодня опробуются несколько более сложных форматов модуляции, основанных на поляризационном мультиплексировании (PM) и квадратурной амплитудной модуляции (QAM, мы рассмотрели ее в главе 9):

- PM-8QAM (2 × 3 бита на символ = 64 состояния);
- PM-16QAM (2 × 4 бита на символ = 256 состояний);
- PM-32QAM (2 × 5 бита на символ = 1024 состояния);
- PM-64QAM (2 × 6 бита на символ = 4096 состояний).

Для генерации таких сложных символов передатчик сигнала должен быть оснащен цифровым сигнальным процессором, вычисляющим сигнал нужной формы в цифровом виде с помощью математических алгоритмов, а также цифроаналоговым преобразователем для представления сигнала в аналоговой форме (напомним, что для перехода на скорость 100 Гбит/с цифровые сигнальные процессоры были добавлены к приемникам, теперь они должны применяться как в приемниках, так и в передатчиках).

Однако увеличение количества состояний символов модулируемого сигнала имеет свои ограничения. Несмотря на повышенную чувствительность когерентного распознавания, увеличение количества состояний требует повышения мощности передатчика, например переход от модуляции PM-QPSK к PM-16QAM (удвоение битовой скорости при той же тактовой частоте) требует повышения мощности передатчика на 7 дБ, что ведет к увеличению нелинейных искажений, которые когерентное распознавание компенсировать не может.

В результате за повышение битовой скорости с помощью более сложных методов модуляции приходится расплачиваться протяженностью отрезков оптических линий, работающих без регенерации оптического сигнала. Так, если отрезок линии без регенерации при модуляции PM-QPSK может достигать 3000 км, то при модуляции PM-16QAM его максимальная длина сокращается до 1500 км, а при модуляции PM-64QAM — до 175 км.

Модуляция PM-16QAM имеет очевидное преимущество перед другими более сложными форматами, так как ее спектр умещается в слоты шириной в 50 ГГц частотного плана DWDM, а практически все оборудование DWDM/OTN поколения 100 Гбит/с работает именно с этим частотным планом. Поэтому повышение скорости со 100 до 200 Гбит/с может быть достигнуто на существующем оборудовании наиболее простым способом, а именно — путем замены транспондеров, поддерживающих модуляцию PM-QPSK, на транспондеры PM-16QAM.

Суперканалы

Сложности с повышением битовой скорости отдельного спектрального канала (необходимость жертвовать расстоянием между регенераторами и повышать мощность передатчика) привели к обращению разработчиков технологии «свыше 100 Гбит/с» к идее суперканала.

Суперканал (super-channel) DWDM — это оптический канал, который использует более одной оптической несущей (волны). Все поднесущие сигнала следуют через сеть по одному и тому же маршруту как единая логическая группа, что позволяет более плотно располагать поднесущие относительно друг друга, чем это было бы возможно при их индивидуальной

маршрутизации (мы опускаем техническое обоснование этого эффекта, которое выходит за рамки темы данной книги).

Идея образования логического канала из нескольких физических не нова, она достаточно успешно применяется при беспроводной передаче, когда радиосигнал образуется путем мультиплексирования большого количества близких частот поднесущих. Логические каналы также успешно применяются в локальных сетях (агрегирование витых пар или оптических волн в версиях 10G и 100G Ethernet — эти технологии рассматриваются в части III данной книги), а также в компьютерах (параллельные шины данных).

Суперканалы DWDM считаются очень перспективным направлением технологии «свыше 100 Гбит/с», которое может привести к появлению каналов 400 Гбит/с и 1 Тбит/с уже в ближайшее время (на момент написания данной книги уже достаточно большое количество производителей оборудования DWDM провели успешные полевые демонстрации таких каналов в действующих сетях).

За счет того, что подканалы суперканала упаковываются в спектральном отношении плотнее, в спектральном диапазоне одной и той же ширины, например в С-диапазоне, можно передавать данные с большей скоростью, чем при использовании обычных каналов. Это преимущество суперканалов иллюстрирует рис. 10.21, где заштрихованные полосы отражают наличие частотных коридоров безопасности между отдельными каналами, предохраняющими сигналы соседних каналов от взаимного наложения и искажения.

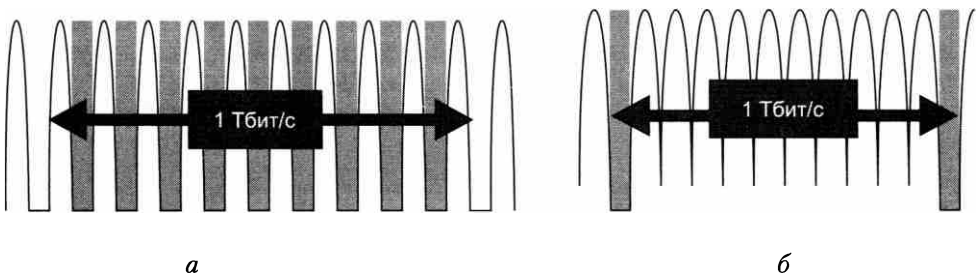


Рис. 10.21. Суперканал имеет более узкий спектр, чем набор индивидуальных каналов: *а* — набор 10 × 100 Гбит/с стандартных каналов; *б* — суперканал 1 Тбит/с с 10 поднесущими 100 Гбит/с (источник: статья «Super-Channels: DWDM Transmission at 100Gb/s and Beyond» компании Infinera)

Преимущество суперканала в более плотной упаковке подканалов можно оценить количественно — для этого применяется такой показатель, как *спектральная эффективность* оборудования. Она измеряется отношением скорости передачи данных к ширине спектра, используемого сигналами данных. Например, при передаче данных со скоростью 100 Гбит/с с помощью сигнала, имеющего спектр 50 ГГц (типичные данные для современного оборудования DWDM для скоростей 100 Гбит/с, не использующего технику суперканалов), спектральная эффективность оборудования равна 2 бит/с/Гц. При наличии суперканалов спектральная плотность в экспериментальном оборудовании достигала 5–7 бит/с/Гц.

Промежуточное оборудование сети DWDM — мультиплексоры, усилители — не имеют дело с подканалами суперканала; они рассматривают суммарный сигнал подканалов как единый сигнал и выделяют ему соответствующую полосу пропускания, необходимую для его поднесущих. Только передатчики и приемники конечных узлов суперканала должны понимать его структуру и уметь ее модулировать и демодулировать.

Предполагается, что различные волны, мультиплексируемые в суперканалы, могут переносить сигналы суперканалов различной скорости, например три суперканала скорости 200 Мбит, два суперканала скорости 500 Мбит/с и два суперканала скорости 1 Тбит/с, при этом их скорости определяются потребностями пользовательских соединений, которые они обслуживают.

Очевидно, что для передачи суперканалов различной скорости необходимо еще одно усовершенствование существующей организации сетей DWDM — переход на *гибкий частотный план*, позволяющий выделять каждому каналу спектральный слот необходимой ширины. Такой частотный план был стандартизован ITU-T в рекомендации G.694-1 редакции 2012 года. Он определяет центр частотного слота с дискретностью 6,25 ГГц, при этом минимальная ширина слота должна быть кратна 12,5 ГГц. Допускается любая комбинация слотов различной ширины при условии, что они не перекрываются.

Выводы

Первичные сети предназначены для создания коммутируемой инфраструктуры, с помощью которой можно достаточно быстро создать постоянные каналы, организующие произвольную топологию.

Цифровые первичные сети PDH позволяют образовывать каналы с пропускной способностью от 64 Кбит/с до 140 Мбит/с, предоставляя своим абонентам скорости четырех уровней иерархии.

Недостатком сетей PDH является невозможность непосредственного выделения данных низкоскоростного канала из данных высокоскоростного канала, если каналы работают на несмежных уровнях иерархии скоростей.

Асинхронность ввода абонентских потоков в кадр SDH обеспечивается благодаря концепции виртуальных контейнеров и системы плавающих указателей, отмечающих начало пользовательских данных в виртуальном контейнере.

В сетях SDH поддерживается большое количество механизмов отказоустойчивости, которые защищают трафик данных на уровне отдельных блоков, портов или соединений: EPS, CP, MSP, SNC-P и MS-SPRing. Наиболее эффективная схема защиты выбирается в зависимости от логической топологии соединений в сети.

Технология WDM/DWDM реализует принципы частотного мультиплексирования для сигналов иной физической природы и на новом уровне иерархии скоростей. Каждый канал WDM/DWDM представляет собой определенный диапазон световых волн, позволяющих переносить данные в аналоговой и цифровой формах, при этом полоса пропускания канала в 25–50–100 ГГц обеспечивает скорости в несколько гигабит в секунду (при передаче дискретных данных).

В ранних системах WDM использовалось небольшое количество спектральных каналов, от 2 до 16. В системах DWDM задействовано уже от 32 до 160 каналов на одном оптическом волокне, что обеспечивает скорости передачи данных для одного волокна до нескольких терабит в секунду.

Современные оптические усилители позволяют удлинить оптический участок линии связи (без преобразования сигнала в электрическую форму) до 700–1000 км.

Для взаимодействия с традиционными оптическими сетями (SDH, Gigabit Ethernet, 10G и 100G Ethernet) в сетях DWDM применяются транспондеры и трансляторы длин волн, которые преобразуют длину волны входного сигнала в длину одной из волн стандартного частотного плана DWDM.

В полностью оптических сетях все операции мультиплексирования и коммутации каналов выполняются над световыми сигналами без их промежуточного преобразования в электрическую форму. Это упрощает и удешевляет сеть.

Технология OTN позволяет более эффективно использовать спектральные каналы сетей DWDM, поддерживая экономные схемы мультиплексирования данных на высоких скоростях.

Мощный механизм коррекции ошибок OTN FEC, использующий самокорректирующиеся коды Рида—Соломона, позволяет улучшить отношение сигнал/шум в спектральных каналах и увеличить расстояние между регенераторами сети.

Повышение скорости сетей OTN/DWDM до 100 Гбит/с потребовало от разработчиков применения технических новшеств, наиболее важными из которых стали:

- модуляция PM-QPSK с 16 состояниями сигнала;
- когерентное распознавание кода;
- применение в приемнике цифрового сигнального процессора;
- использование более эффективных кодов FEC.

Работы над дальнейшим повышением скорости сетей OTN/DWDM привели к появлению техники суперканалов с несколькими волновыми подканалами. Суперканалы обеспечивают более высокую спектральную эффективность канала за счет более плотного размещения подканалов относительно друг друга.

Контрольные вопросы

1. Каким образом компенсируется отсутствие синхронности трибутарных потоков в технологии SDH?
2. В чем отличие схем защиты 1 + 1 и 1 : 1? Варианты ответов:
   а) в схеме 1 + 1 два потока мультиплексируются в один, а в схеме 1 : 1 — нет;
   б) схема 1 + 1 говорит о том, что резервный элемент выполняет те же функции, что и основной, а в схеме 1 : 1 резервный элемент простаивает до момента выхода из строя основного;
   в) схема 1 + 1 используется для защиты портов, а схема 1 : 1 — для защиты путей трафика.
3. Почему протокол GFP в режиме GFP-F не использует пустые кадры для выравнивания скоростей?
4. Что общего между первичными сетями FDM и DWDM?
5. Реконфигурируемый оптический мультиплексор ввода-вывода называется бесцветным ненаправленным, если:
   а) может вывести любую волну из любого направления в любой пользовательский порт ввода-вывода путем программного реконфигурирования;
   б) позволяет обслуживать любой запрос на маршрутизацию волны независимо от того, какие запросы на маршрутизацию других волн он уже обслуживает.

Часть III

Локальные вычислительные сети

- Глава 11. Технология локальных сетей на разделяемой среде
- Глава 12. Коммутируемые сети Ethernet
- Глава 13. Отказоустойчивость и виртуализация локальных сетей

Локальные сети являются неотъемлемой частью любой современной компьютерной сети. Если мы рассмотрим структуру глобальной сети, например Интернета или крупной корпоративной сети, то обнаружим, что практически все информационные ресурсы этой сети сосредоточены в локальных сетях, а глобальная сеть является транспортом, который соединяет многочисленные локальные сети.

Технологии локальных сетей прошли большой путь. Практически во всех технологиях 80-х годов использовалась *разделяемая среда* как удобное и экономичное средство объединения компьютеров на физическом уровне. С середины 90-х в локальных сетях стали применяться *коммутируемые* версии технологий. Отказ от разделяемой среды позволил повысить производительность и масштабируемость локальных сетей. Преимуществом коммутируемых локальных сетей является также возможность логической структуризации сети с разделением ее на отдельные сегменты, называемые виртуальными локальными сетями.

Переход к коммутируемым локальным средам сопровождался победой одной технологии, а именно технологии Ethernet. Остальные технологии, такие как Token Ring и FDDI, остались в прошлом, несмотря даже на то, что они обладали хорошими техническими характеристиками и имели многочисленных пользователей.

Неизвестно, что больше повлияло на такую ситуацию, то ли предельная простота технологии, а значит, и низкая стоимость оборудования Ethernet и его эксплуатации, то ли удачное название, то ли просто необыкновенное везение, как считает изобретатель этой технологии Роберт Меткалф, состоявшее в том, что «каждый раз, когда появлялось что-то на замену Ethernet, люди, ответственные за продвижение новой технологии, снова выбирали для нее название Ethernet», но факт остается фактом — локальные сети стали однородными коммутируемыми сетями Ethernet.

В локальных сетях изменился не только принцип использования среды. Быстро растет верхний предел информационной скорости протоколов локальных сетей. Сегодня иерархия скоростей локальных сетей соответствует иерархии скоростей первичных сетей — от 10 Мбит/с до 100 Гбит/с. Это дает возможность строить на этих технологиях не только локальные сети, но и сети мегаполисов, а также эффективно соединять локальные и глобальные сети без необходимости согласования их скоростей. Развитие локальных сетей идет и в направлении «миниатюризации» — появился новый тип сетей — *персональные сети* (Personal Area Network, PAN), которые объединяют электронные устройства одного пользователя в радиусе нескольких десятков метров.

В главе 11 рассматриваются технологии локальных сетей на разделяемой среде: основное внимание уделено классическим вариантам Ethernet со скоростью 10 Мбит/с на коаксиале и витой паре.

Если проводные технологии локальных сетей на разделяемой среде интересны сегодня в основном в теоретическом плане (для понимания истоков и динамики развития современных технологий), то беспроводные технологии локальных сетей на основе разделяемой среды по-прежнему актуальны и, по всей видимости, останутся таковыми в обозримом будущем, так как радиоэфир является разделяемой средой по своей природе. Мы рассмотрим две наиболее популярные технологии этого семейства — IEEE 802.11 (LAN) и Bluetooth (PAN).

Глава 12 посвящена коммутируемым локальным сетям. В ней рассматриваются основные принципы работы таких сетей: алгоритм функционирования коммутатора локальной сети, дуплексные версии протоколов локальных сетей, особенности реализации коммутаторов локальных сетей. Здесь также приводятся описания скоростных версий Ethernet: Fast Ethernet, Gigabit Ethernet, 10G и100G Ethernet, разработанных исключительно для применения в коммутируемых сетях.

В главе 13 изучаются вопросы обеспечения отказоустойчивости локальных коммутируемых сетей с помощью протокола покрывающего дерева (STP), а также техника виртуальных локальных сетей (VLAN), позволяющая быстро и эффективно выполнять логическую структуризацию сети.

ГЛАВА 11 Технологии локальных сетей на разделяемой среде

Общая характеристика протоколов локальных сетей на разделяемой среде

Сегодня технологии локальных сетей на разделяемой среде применяются только в беспроводных локальных сетях (называемых также сетями Wi-Fi). В проводных же локальных сетях уже довольно давно, с середины 1990-х годов, разделяемая среда не используется из-за плохой масштабируемости такого подхода. И хотя в стандартах единственной выжившей технологии локальных проводных сетей — Ethernet — вариант работы на разделяемой среде все еще описан, он разрешен к применению только для низко- и среднескоростных версий Ethernet, но не для скоростей 10 и 100 Гбит/с.

Тем не менее мы включили в книгу описание основных идей и характеристик Ethernet и других технологий на разделяемой проводной среде, так как это помогает понять особенности техники применения разделяемой среды, что полезно при разработке новых технологий беспроводных сетей, где эта техника является естественной. Кроме того, знания истории развития технологии помогает лучше понять некоторые ее унаследованные черты, такие как, например, размер и формат кадра Ethernet, сохранившиеся и в современных коммутируемых версиях Ethernet.

Стандартная топология и разделяемая среда

Основная цель, которую ставили перед собой разработчики первых локальных сетей во второй половине 70-х годов, заключалась в нахождении простого и дешевого решения для объединения в вычислительную сеть нескольких десятков компьютеров, находящихся в пределах одного здания. Решение должно было быть недорогим, поскольку компьютеры, объединявшиеся в сеть, были недороги — появившиеся и быстро распространявшиеся тогда мини-компьютеры стоили 10 000–20 000 долларов (это было действительно очень дешево по сравнению со стоимостью мейнфреймов). Количество их в одной организации было небольшим, поэтому предел в несколько десятков компьютеров представлялся вполне достаточным для практически любой локальной сети. Задача связи локальных сетей в глобальные не была первоочередной, поэтому практически все технологии локальных сетей ее игнорировали.

Для упрощения и соответственно удешевления аппаратных и программных решений разработчики первых локальных сетей остановились на совместном использовании **общей среды передачи данных**.

Этот метод связи компьютеров впервые был опробован при создании радиосети ALOHA Гавайского университета в начале семидесятых под руководством Нормана Абрамсона. Радиоканал определенного диапазона частот естественным образом является общей средой для всех передатчиков, использующих частоты этого диапазона для кодирования данных. Сеть ALOHA работала по методу случайного доступа, когда каждый узел мог начать передачу пакета в любой момент времени. Если после этого он не дожидался подтверждения приема в течение определенного тайм-аута, он посылал этот пакет снова. Общим был радиоканал с несущей частотой 400 МГц и полосой 40 КГц, что обеспечивало передачу данных со скоростью 9600 бит/с.

Немного позже Роберт Меткалф (Robert Metcalfe) повторил идею разделяемой среды уже для проводного варианта технологии LAN. Непрерывный сегмент коаксиального кабеля стал аналогом общей радиосреды. Все компьютеры присоединялись к этому сегменту кабеля по схеме монтажного ИЛИ, поэтому при передаче сигналов одним из передатчиков все приемники получали один и тот же сигнал, как и при использовании радиоволн. На рис. 11.1 представлено начало служебной записки Роберта Меткалфа, написанной 22 мая 1973 года, с наброском разделяемой среды на коаксиальном кабеле, где эта среда названа «a cable-tree ether», что можно приблизительно перевести как «древовидный кабельный эфир».

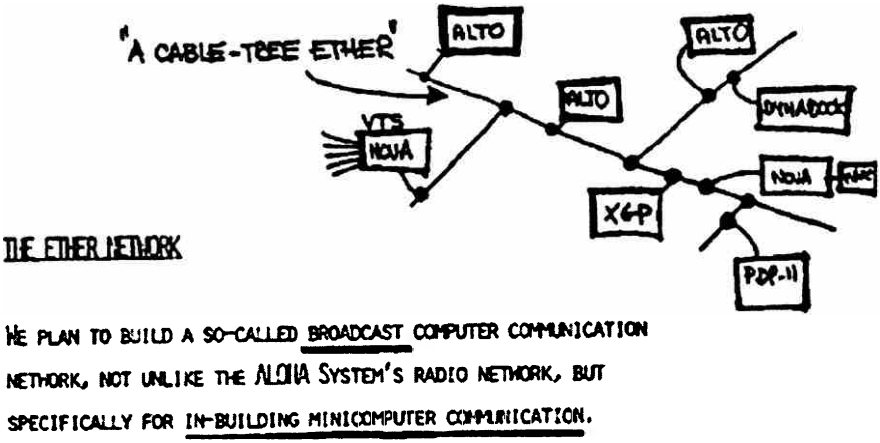


Рис. 11.1. Рисунок Роберта Меткалфа с иллюстрацией идеи эмуляции разделяемого радиоэфира с помощью коаксиального кабеля

В технологиях Token Ring и FDDI тот факт, что компьютеры используют разделяемую среду, не так очевиден, как в случае Ethernet. Физическая топология этих сетей — кольцо, каждый узел соединяется кабелем с двумя соседними узлами (рис. 11.2). Однако эти отрезки кабеля также являются разделяемыми, так как в каждый момент времени только один компьютер может задействовать кольцо для передачи своих пакетов.

Простые **стандартные топологии физических связей** (звезда у коаксиального кабеля Ethernet и кольцо у Token Ring и FDDI) обеспечивают простоту разделения кабельной среды.

Рис. 11.2. Разделяемая среда в кольцевых топологиях

Использование разделяемых сред позволяет *упростить* логику работы узлов сети. Действительно, поскольку в каждый момент времени выполняется только одна передача, отпадает необходимость в буферизации кадров в транзитных узлах, и как следствие — в самих транзитных узлах. Соответственно отпадает необходимость в сложных процедурах управления потоком и борьбы с перегрузками.

Основной недостаток разделяемой среды — плохая масштабируемость. Этот недостаток является принципиальным, так как независимо от метода доступа к среде ее пропускная способность делится между всеми узлами сети. Здесь применимо положение теории очередей, которое мы изучали в главе 6: как только коэффициент использования общей среды превышает определенный порог, очереди к среде начинают расти нелинейно и сеть становится практически неработоспособной. Значение порога зависит от метода доступа. Так, в сетях ALOHA это значение является крайне низким — всего около 18 %, в сетях Ethernet — около 30, а в сетях Token Ring и FDDI оно возросло до 60–70 %.

Локальные сети, являясь пакетными сетями, используют принцип временно́го мультиплексирования, то есть разделяют передающую среду во времени. Алгоритм *управления доступом к среде* является одной из важнейших характеристик любой технологии LAN на разделяемой среде, в значительно большей степени определяя ее облик, чем метод кодирования сигналов или формат кадра. В технологии Ethernet в качестве алгоритма разделения среды применяется *метод случайного доступа*. И хотя его трудно назвать совершенным — при росте нагрузки полезная пропускная способность сети резко падает — он благодаря своей простоте стал основой успеха технологии Ethernet в 80-е годы. Технологии Token Ring и FDDI используют *метод маркерного доступа*, основанный на передаче от узла к узлу особого кадра — маркера (токена) доступа. При этом только узел, владеющий маркером доступа, имеет право доступа к разделяемому кольцу. Более детерминированный характер доступа технологий Token Ring и FDDI предопределил более эффективное использование разделяемой среды, чем у технологии Ethernet, но одновременно и усложнил оборудование.

Отказ от разделяемой среды привел к исчезновению такого важного компонента технологии локальных сетей, как метод доступа. В принципе, коммутатор локальной сети работает так же, как и обобщенный коммутатор сети с коммутацией пакетов, рассмотренный в главе 2. Поэтому с распространением коммутаторов стали стираться различия между технологиями локальных сетей, так как в сети, где все связи между узлами являются индивидуальными, и коммутируемая версия Ethernet, и коммутируемая версия Token

Ring работают весьма схоже, различаются только форматы кадров этих технологий. Это обстоятельство, возможно, и имел в виду Роберт Меткалф, когда говорил об удачливости Ethernet, — работа коммутируемых локальных сетей Ethernet существенно отличается от работы Ethernet на разделяемой среде, так что ее можно считать новой технологией со старым названием. Хотя, с другой стороны, формат кадра Ethernet сохранился, так что это дает формальный (хотя и несколько условный) повод считать ее той же самой технологией.

Стандартизация протоколов локальных сетей

Каждая из технологий локальных сетей первоначально появлялась как фирменная технология; так, например, технология Ethernet «появилась на свет» в компании Xerox, а за технологией Token Ring стояла компания IBM. Первые стандарты технологий локальных сетей также были фирменными, что было, естественно, не очень удобно как для пользователей, так и для компаний-производителей сетевого оборудования.

Для исправления ситуации в 1980 году в институте IEEE был организован комитет 802 по стандартизации технологий LAN. Результатом работы комитета IEEE 802 стало принятие семейства стандартов IEEE 802.x, содержащих рекомендации по проектированию нижних уровней локальных сетей. Эти стандарты базировались на обобщении популярных фирменных стандартов, в частности Ethernet и Token Ring.

Комитет IEEE 802 и сегодня является основным международным органом, разрабатывающим стандарты технологий локальных сетей, в том числе стандарты коммутируемых локальных сетей, а также беспроводных локальных сетей на разделяемой среде.

Помимо IEEE в работе по стандартизации протоколов LAN принимали и принимают участие и другие организации. Так, для сетей, работающих на оптоволокне, институтом ANSI был разработан стандарт FDDI, обеспечивающий скорость передачи данных 100 Мбит/с. Это был первый протокол LAN, который достиг такой скорости, в 10 раз превысив скорость технологии Ethernet.

Структуру стандартов IEEE 802 иллюстрирует рис. 11.3.

Стандарты IEEE 802 описывают функции, которые можно отнести к функциям физического и канального уровней модели OSI. Как видно из рисунка, эти стандарты имеют и общие, и индивидуальные для всех технологий части.

Общую группу стандартов составляют *стандарты рабочей группы 802.1*. Эти стандарты описывают наиболее высокоуровневые функции локальных сетей. Так, в документах 802.1 даются общие определения локальных сетей и их свойств, показана связь трех уровней модели IEEE 802 с моделью OSI. Наиболее важным в настоящее время является стандарт 802.1D, описывающий логику работы прозрачного моста, которая лежит в основе любого современного коммутатора Ethernet (и лежала бы в основе коммутатора Token Ring или FDDI, если бы они сохранились до наших дней).

Набор стандартов, разработанных рабочей группой 802.1, продолжает расти, в настоящее время это наиболее активный подкомитет комитета 802. Например, этот комитет стандартизовал технологию виртуальных локальных сетей, также он занимается стандартизацией технологий, известных под общим названием Carrier Ethernet.

Каждая из рабочих групп 802.3, 802.4, 802.5 и т. д. была ответственна за стандартизацию конкретной технологии, например группа 802.3 занималась технологией Ethernet, группа

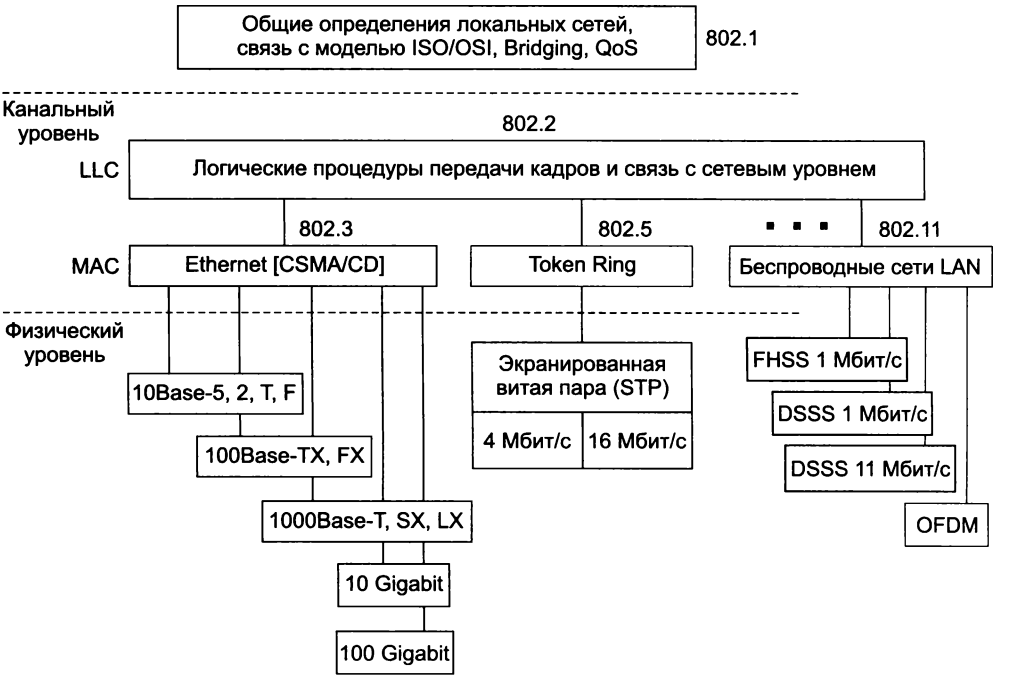


Рис. 11.3. Структура стандартов IEEE 802.x

802.4 — технологией ArcNet, группа 802.5 — технологией Token Ring, группа 802.11 — технологией беспроводных локальных сетей. Сегодня из этих групп активными остались только 802.3 и 802.11 (существуют также и другие активные группы комитета IEEE 802, но они не занимаются технологиями локальных сетей).

Стандарты этих рабочих групп описывают как физический уровень (или несколько возможных физических уровней), так и канальный уровень конкретной технологии (последний включает описание метода доступа, используемого технологией).

Основу стандарта 802.3 составила технология экспериментальной сети Ethernet Network, которую фирма Xerox разработала и реализовала в 1975 году. В 1980 году фирмы DEC, Intel и Xerox (сокращенно — DIX) совместно разработали и опубликовали стандарт Ethernet версии II для сети, построенной на основе коаксиального кабеля. Эту последнюю версию фирменного стандарта Ethernet называют стандартом Ethernet DIX, или Ethernet II. На базе стандарта Ethernet DIX был разработан стандарт IEEE 802.3, который во многом совпадает со своим предшественником.

Однако, как видно из рис. 11.3, помимо индивидуальных для каждой технологии уровней существует общий уровень, который был стандартизован рабочей группой 802.2.

Появление этого уровня связано с тем, что комитет 802 разделил функции канального уровня модели OSI на два уровня:

- управления логическим каналом (Logical Link Control, LLC);
- управления доступом к среде (Media Access Control, MAC).

Основными функциями уровня MAC являются:

- обеспечение доступа к разделяемой среде;
- передача кадров между конечными узлами посредством функций и устройств физического уровня.

Если уровень MAC специфичен для каждой технологии и отражает различия в методах доступа к разделяемой среде, то уровень LLC представляет собой обобщение функций разных технологий по обеспечению передачи кадра с различными требованиями к надежности.

Логика образования общего для всех технологий уровня LLC заключается в следующем: после того как узел сети получил доступ к среде в соответствии с алгоритмом, специфическим для конкретной технологии, дальнейшие действия узла или узлов по обеспечению надежной передачи кадров не зависят от этой технологии.

Так как в зависимости от требований приложения может понадобиться разная степень надежности передачи, то рабочая группа 802.2 определила три типа услуг:

- **Услуга LLC1** — это услуга *без установления соединения и без подтверждения получения данных*.
- **Услуга LLC2** дает пользователю возможность установить *логическое соединение* перед началом передачи любого блока данных, и если это требуется, выполнить *процедуры восстановления* после ошибок и упорядочивание потока блоков в рамках установленного соединения.
- **Услуга LLC3** — это услуга *без установления соединения, но с подтверждением получения данных*.

Какой из трех режимов работы уровня LLC будет использован, зависит от требований протокола верхнего уровня.

Нужно сказать, что на практике идея обобщения функций обеспечения надежной передачи кадров в общем уровне LLC не оправдала себя. Технология Ethernet в версии DIX изначально функционировала в наиболее простом дейтаграммном режиме — в результате оборудование Ethernet и после опубликования стандарта IEEE 802.2 продолжало поддерживать только этот режим работы, который формально является режимом LLC1. В то же время оборудование сетей Token Ring, которое изначально поддерживало режимы LLC2 и LLC3, также продолжало поддерживать эти режимы и никогда не поддерживало режим LLC1. В настоящее время задача обеспечения надежной передачи данных в наибольшей мере возлагается на протокол TCP, в котором реализован механизм контроля потока на основе концепции скользящего окна. Читайте об этом в разделе «Методы квитирования» главы 16.

Ethernet со скоростью 10 Мбит/с на разделяемой среде

MAC-адреса

На уровне MAC, который обеспечивает доступ к среде и передачу кадра, для идентификации сетевых интерфейсов узлов сети используются регламентированные стандартом

IEEE 802.3 уникальные 6-байтовые адреса, называемые **MAC-адресами**. Обычно MAC-адрес записывают в виде шести пар шестнадцатеричных цифр, разделенных дефисами или двоеточиями, например 11-A0-17-3D-BC-01. Каждый сетевой адаптер имеет по крайней мере один MAC-адрес.

Помимо отдельных интерфейсов MAC-адрес может определять группу интерфейсов или даже все интерфейсы сети. Первый (младший) бит старшего байта адреса назначения — это признак того, является адрес индивидуальным или групповым. Если он равен 0, то адрес является **индивидуальным**, то есть идентифицирует один сетевой интерфейс, а если 1, то **групповым**. Групповой адрес связан только с интерфейсами, сконфигурированными (вручную или автоматически по запросу вышележащего уровня) как члены группы, номер которой указан в групповом адресе. Если сетевой интерфейс включен в группу, то наряду с уникальным MAC-адресом с ним ассоциируется еще один адрес — групповой. В частном случае, если групповой адрес состоит из всех единиц, то есть имеет шестнадцатеричное представление 0xFFFFFFFFFFFF, он идентифицирует все узлы сети и называется **широковещательным**.

Второй бит старшего байта адреса определяет способ назначения адреса — **централизованный** или **локальный**. Если этот бит равен 0 (что бывает почти всегда в стандартной аппаратуре Ethernet), это говорит о том, что адрес назначен централизованно по правилам IEEE 802.

ВНИМАНИЕ

В стандартах IEEE Ethernet младший бит байта изображается в самой левой позиции поля, а старший бит — в самой правой. Этот нестандартный способ отображения порядка следования битов в байте соответствует порядку передачи битов в линию связи передатчиком Ethernet (первым передается младший бит). В стандартах других организаций, например RFC IETF, ITU-T, ISO, используется традиционное представление байта, когда младший бит считается самым правым битом байта, а старший — самым левым. При этом порядок следования байтов остается традиционным. Поэтому при чтении стандартов, опубликованных этими организациями, а также чтении данных, отображаемых на экране операционной системой или анализатором протоколов, значения каждого байта в кадре Ethernet нужно зеркально отобразить, чтобы получить представление о значении разрядов этого байта в соответствии с документами IEEE. Например, групповой адрес, имеющий в нотации IEEE вид 1000 0000 0000 0000 1010 0111 1111 0000 0000 0000 0000 0000 или в шестнадцатеричной записи 80-00-A7-F0-00-00, будет, скорее всего, отображен анализатором протоколов в традиционном виде как 01-00-5E-0F-00-00.

Комитет IEEE распределяет между производителями оборудования так называемые **организационно уникальные идентификаторы** (Organizationally Unique Identifier, OUI). Каждый производитель помещает выделенный ему идентификатор в три старших байта адреса (например, идентификатор 0x0020AF определяет компанию 3COM, а 0x00000C — Cisco). За уникальность младших трех байтов адреса отвечает производитель оборудования. Двадцать четыре бита, отводимые производителю для адресации интерфейсов его продукции, позволяют выпустить примерно 16 миллионов интерфейсов под одним идентификатором организации. Уникальность централизованно распределяемых адресов распространяется на все основные технологии локальных сетей — Ethernet, Token Ring, FDDI и т. д. Локальные адреса назначаются администратором сети, в обязанности которого входит обеспечение их уникальности.

Сетевые адаптеры Ethernet могут также работать в так называемом **«неразборчивом» режиме** (promiscuous mode), когда они захватывают все кадры, поступающие на интерфейс, независимо от их MAC-адресов назначения. Обычно такой режим используется для мониторинга трафика, когда захваченные кадры изучаются затем для нахождения причины некорректного поведения некоторого узла или отладки нового протокола.

Форматы кадров технологии Ethernet

Существует несколько стандартов формата кадра Ethernet. На практике в оборудовании Ethernet используется только один формат кадра, а именно — кадр Ethernet DIX, который иногда называют кадром Ethernet II по номеру последнего стандарта DIX. Этот формат представлен на рис. 11.4.

6 байт	6 байт	2 байта	46–1500 байт	4 байта
DA	SA	T	Данные	FCS

Рис. 11.4. Формат кадра Ethernet DIX (II)

Первые два поля заголовка отведены под адреса:

- **DA (Destination Address)** — MAC-адрес узла назначения;
- **SA (Source Address)** — MAC-адрес узла отправителя.

  Для доставки кадра достаточно одного адреса — адреса назначения; адрес источника помещается в кадр для того, чтобы узел, получивший кадр, знал, от кого пришел кадр и кому нужно на него ответить. Принятие решения об ответе не входит в компетенцию протокола Ethernet, это дело протоколов верхних уровней, Ethernet лишь выполнит такое действие, если с сетевого уровня поступит соответствующее указание.
- **Поле T (Type**, или **EtherType)** содержит условный код протокола верхнего уровня, данные которого находятся в поле данных кадра, например шестнадцатеричное значение 08-00 соответствует протоколу IP. Это поле требуется для поддержки интерфейсных функций мультиплексирования и демультиплексирования кадров при взаимодействии с протоколами верхних уровней.
- **Поле данных** может содержать от 46 до 1500 байт. Если длина пользовательских данных меньше 46 байт, то это поле дополняется до минимального размера байтами заполнения. Эта операция требуется для корректной работы метода доступа Ethernet (он рассматривается в следующем разделе).
- **Поле контрольной последовательности кадра** (Frame Check Sequence, FCS) состоит из 4 байт контрольной суммы. Это значение вычисляется по алгоритму CRC-32.

Кадр Ethernet DIX (II) не отражает разделения канального уровня Ethernet на уровни MAC и LLC: его поля поддерживают функции обоих уровней, например интерфейсные функции поля *T* относятся к функциям уровня LLC, в то время как все остальные поля поддерживают функции уровня MAC.

Существуют еще три стандартных формата кадра Ethernet:

- Кадр 802.3/LLC является стандартом комитета IEEE 802 и построен в соответствии с принятым разбиением функций канального уровня на уровни MAC и LLC. Поэтому результирующий кадр является вложением кадра LLC, определяемого стандартом 802.2, в кадр MAC, определяемый стандартом 802.3.
- Кадр **Raw 802.3**, или **Novell 802.3**, появился в результате усилий компании Novell по ускорению разработки своего стека протоколов в сетях Ethernet.
- Кадр **Ethernet SNAP** стал результатом деятельности комитета 802.2 по приведению предыдущих форматов кадров к некоторому общему стандарту и приданию кадру необходимой гибкости для учета в будущем возможностей добавления полей или изменения их назначения.

Как уже отмечалось, в настоящее время оборудованием Ethernet используются только кадры Ethernet DIX (II). Остальные форматы кадров, в том числе кадр 802.3/LLC, по-прежнему формально являющийся стандартным, вышли из употребления из-за более сложного формата, который оказался не нужен в условиях существования единой технологии канального уровня.

(S) *Форматы кадров Ethernet*

Доступ к среде и передача данных

Метод доступа, используемый в сетях Ethernet на разделяемой проводной среде[1], носит название CSMA/CD (Carrier Sense Multiple Access with Collision Detection — прослушивание несущей частоты с множественным доступом и распознаванием коллизий). Название метода достаточно хорошо описывает его особенности.

Все компьютеры в сети на разделяемой среде имеют возможность немедленно (с учетом задержки распространения сигнала в физической среде) получить данные, которые любой из компьютеров начал передавать в общую среду. Говорят, что среда, к которой подключены все станции, работает в режиме **коллективного доступа** (Multiply Access, MA).

Чтобы получить возможность передавать кадр, интерфейс-отправитель должен убедиться, что разделяемая среда свободна. Это достигается прослушиванием основной гармоники сигнала, которая еще называется **несущей частотой** (Carrier Sense, CS).

Признаком «незанятости» среды является отсутствие на ней несущей частоты, которая при манчестерском способе кодирования равна 5–10 МГц в зависимости от последовательности единиц и нулей, передаваемых в данный момент.

Если среда свободна, то узел имеет право начать передачу кадра. В примере, показанном на рис. 11.5, узел 1 обнаружил, что среда свободна, и начал передавать свой кадр. В коаксиальном кабеле сигналы передатчика узла 1 распространяются в обе стороны, так что их получают все узлы сети. Кадр данных всегда сопровождается **преамбулой**, которая состоит из 7 байт, каждый из которых имеет значение 10101010, и 8-го байта, равного 10101011. Последний байт носит название **ограничителя начала кадра**. Преамбула нужна для вхождения приемника в побитовую и побайтовую синхронизацию с передатчиком. Наличие

[1] В беспроводных сетях Ethernet применяется другой метод доступа, известный как CSMA/CA. Этот метод рассматривается далее в разделе «Беспроводные локальные сети IEEE 802.11».

двух последовательных единиц говорит приемнику о том, что преамбула закончилась и следующий бит является началом кадра.

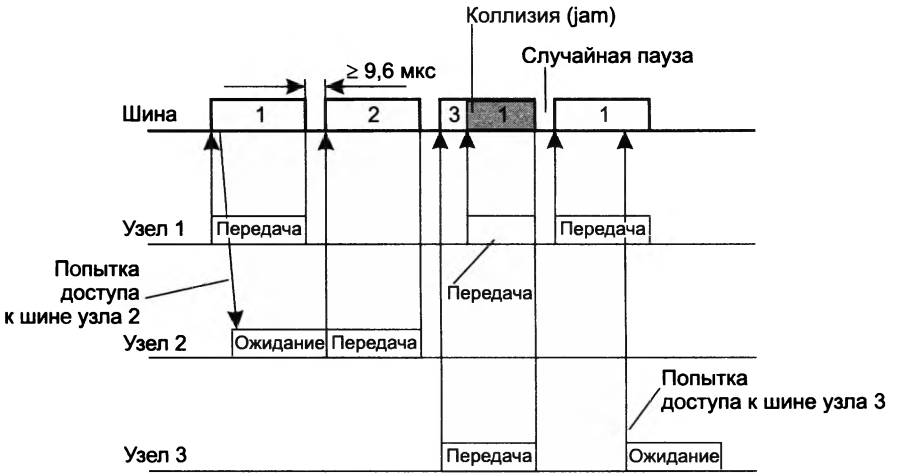


Рис. 11.5. Метод случайного доступа CSMA/CD

Все станции, подключенные к кабелю, начинают записывать байты передаваемого кадра в свои внутренние буферы. Первые 6 байт кадра содержат адрес назначения. Та станция, которая узнает собственный адрес в заголовке кадра, продолжает записывать его содержимое в свой внутренний буфер, а остальные станции на этом прием кадра прекращают. Станция назначения обрабатывает полученные данные и передает их вверх по своему стеку. Кадр Ethernet содержит не только адрес назначения, но и адрес источника данных, поэтому станция-получатель знает, кому нужно послать ответ.

Узел 2 во время передачи кадра узлом 1 также попытался начать передачу своего кадра, однако обнаружив, что среда занята — на ней присутствует несущая частота, — вынужден ждать, пока узел 1 не прекратит передачу кадра.

После окончания передачи кадра все узлы сети обязаны выдержать технологическую паузу, равную **межпакетному интервалу** (Inter Packet Gap, IPG) в 9,6 мкс. Эта пауза нужна для приведения сетевых адаптеров в исходное состояние, а также для предотвращения монопольного захвата среды одной станцией. После окончания технологической паузы узлы имеют право начать передачу своего кадра, так как среда свободна. В приведенном примере узел 2 дождался окончания передачи кадра узлом 1, сделал паузу в 9,6 мкс и начал передачу своего кадра.

Возникновение коллизии

Механизм прослушивания среды и пауза между кадрами не гарантируют исключения ситуации, когда две или более станции одновременно решают, что среда свободна, и начинают передавать свои кадры. Говорят, что при этом происходит **коллизия**, так как содержимое обоих кадров сталкивается в общем кабеле и происходит искажение информации.

Коллизия — это нормальная ситуация в работе сетей Ethernet на разделяемой среде. В примере на рис. 11.6 коллизию породила одновременная передача данных узлами 3 и 1.

Для возникновения коллизии не обязательно, чтобы несколько станций начали передачу *абсолютно* одновременно, такая ситуация маловероятна. Более вероятна ситуация, когда один узел начинает передачу, а через некоторое (короткое) время другой узел, проверив среду и не обнаружив несущую (сигналы первого узла еще не успели до него дойти), начинает передачу своего кадра. Таким образом, возникновение коллизии является следствием распределения узлов сети в пространстве.

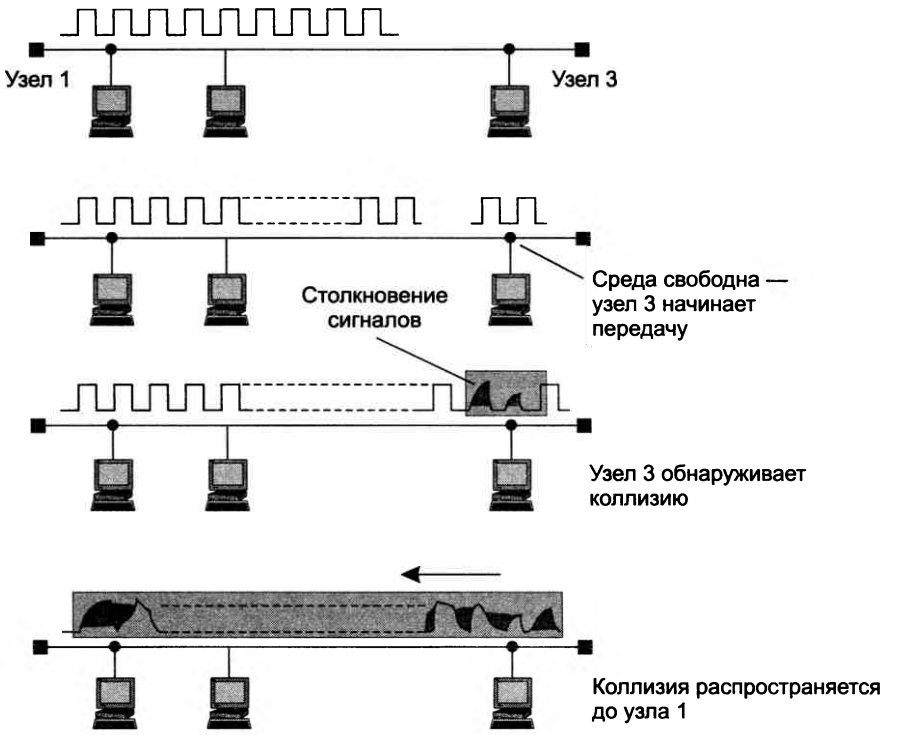


Рис. 11.6. Схема возникновения и распространения коллизии

Чтобы корректно обработать коллизию, все станции одновременно наблюдают за возникающими на кабеле сигналами. Если передаваемые и наблюдаемые сигналы отличаются, то фиксируется факт **обнаружения коллизии** (Collision Detection, CD). Для повышения вероятности скорейшего обнаружения коллизии всеми станциями сети станция, которая обнаружила коллизию, прерывает передачу своего кадра (в произвольном месте, возможно, и не на границе байта) и усугубляет коллизию посылкой в сеть специальной последовательности из 32 бит, называемой **jam-последовательностью**.

После этого обнаружившая коллизию передающая станция обязана прекратить передачу и сделать паузу в течение короткого случайного интервала времени. Затем она может снова предпринять попытку захвата среды и передачи кадра. Случайная пауза выбирается по следующему алгоритму:

$$\text{Пауза} = L \times (\text{интервал отсрочки}).$$

В технологии Ethernet **интервал отсрочки** выбран равным значению 512 битовых интервалов. Битовый интервал соответствует времени между появлением двух последовательных битов данных на кабеле; для скорости 10 Мбит/с величина битового интервала равна 0,1 мкс, или 100 нс.

L представляет собой целое число, выбранное с равной вероятностью из диапазона $[0, 2^N]$, где N — номер повторной попытки передачи данного кадра: 1, 2, ..., 10. После 10-й попытки интервал, из которого выбирается пауза, не увеличивается.

Таким образом, случайная пауза в технологии Ethernet может принимать значения от 0 до 52,4 мс ($51{,}2$ мкс $\times 2^{10}$).

Если 16 последовательных попыток передачи кадра вызывают коллизию, то передатчик должен прекратить попытки и отбросить этот кадр. Описанный алгоритм носит название **усеченного экспоненциального двоичного алгоритма отсрочки**.

Поведение сети Ethernet при значительной нагрузке, когда коэффициент использования среды растет и начинает приближаться к 1, в целом соответствует графикам, которые были приведены в главе 6 при анализе модели теории очередей. Однако рост времени ожидания освобождения среды в сетях Ethernet начинается раньше, чем в данной модели. Это происходит из-за того, что в ней не учитывается такая важная особенность Ethernet, как коллизии.

Администраторы сетей Ethernet на разделяемой среде руководствовались простым эмпирическим правилом — коэффициент использования среды не должен превышать 30 %. Для поддержки чувствительного к задержкам трафика сети Ethernet (и другие сети на разделяемой среде) могут применять только один метод поддержания характеристик QoS — *недогруженный режим работы*.

Время оборота и распознавание коллизий

Надежное распознавание коллизий всеми станциями сети является необходимым условием корректной работы сети Ethernet. Если какая-либо передающая станция не распознает коллизию и решит, что кадр данных передан ею верно, этот кадр будет утерян. Из-за наложения сигналов при коллизии информация кадра исказится, и он будет отбракован принимающей станцией из-за несовпадения контрольной суммы. Скорее всего, не дошедшие до получателя данные будут повторно переданы каким-либо протоколом верхнего уровня, например транспортным или прикладным, работающим с установлением соединения, либо протоколом LLC, если он работает в режиме LLC2. Однако повторная передача сообщения протоколами верхних уровней произойдет гораздо позже (иногда по прошествии нескольких секунд), чем повторная передача средствами сети Ethernet, работающей с микросекундными интервалами. Поэтому если коллизии не будут надежно распознаваться узлами сети Ethernet, то это приведет к заметному снижению полезной пропускной способности сети.

Для надежного распознавания коллизий должно выполняться следующее соотношение:

$$T_{\min} \geq \text{RTT}.$$

Здесь $T_{\min}$ — время передачи кадра минимальной длины, а RTT — *время оборота*, то есть время, за которое сигнал, посланный некоторой станцией сети, доходит то точки колли-

зии, а затем возвращается к станции-отправителю в уже искаженной коллизией форме. В худшем случае сигнал должен пройти дважды между наиболее удаленными друг от друга станциями сети.

При выполнении этого условия передающая станция должна успеть обнаружить коллизию, которую вызвал переданный ею кадр, еще до того, как она закончит передачу этого кадра.

Очевидно, что выполнение этого условия зависит, с одной стороны, от минимальной длины кадра и скорости передачи данных протокола, с другой — от длины кабельной системы сети и скорости распространения сигнала в кабеле (для разных типов кабеля эта скорость несколько отличается).

Все параметры протокола Ethernet, в том числе минимальный размер кадра, подобраны таким образом, чтобы при нормальной работе сети коллизии четко распознавались.

Так, стандарт Ethernet определяет минимальную длину поля данных кадра в 46 байт (что вместе со служебными полями дает минимальную длину кадра 64 байт, а вместе с преамбулой — 72 байт, или 576 бит). Отсюда может быть вычислено ограничение на расстояние между станциями. В стандарте Ethernet 10 Мбит/с время передачи кадра минимальной длины равно 575 битовых интервалов, следовательно, время оборота должно быть меньше 57,5 мкс. Расстояние, которое сигнал может пройти за это время, зависит от типа кабеля и для толстого коаксиального кабеля равно примерно 13 280 метров. Учитывая, что за время 57,5 мкс сигнал должен пройти по линии связи дважды, расстояние между двумя узлами не должно быть больше 6635 м. В стандарте величина этого расстояния выбрана равной 2500 м, что существенно меньше. Это объясняется тем, что повторители, которые нужны для соединения 5 сегментов кабеля, вносят задержки в распространение сигнала.

Описанные соображения объясняют выбор минимальной длины поля данных кадра в 46 байт. Уменьшение этого значения до 0 привело бы к значительному сокращению максимальной длины сети.

Требование $T_{min} \geq RTT$ имеет одно интересное следствие: чем выше скорость протокола, тем меньше должна быть максимальная длина сети. Поэтому для Ethernet на разделяемой среде при скорости в 100 Мбит/с максимальная длина сети пропорционально уменьшается до 250 м, а при скорости в 1 Гбит/с — до 25 м. Эта зависимость наряду с резким ростом задержек при повышении загрузки сети говорит еще об одном коренном недостатке метода доступа CSMA/CD.

Физические стандарты 10M Ethernet

При первоначальной стандартизации технологии Ethernet рабочей группой IEEE 802.3 был выбран вариант Ethernet на «толстом» коаксиальном кабеле, который получил название 10Base-5.

Число 10 этом названии обозначает номинальную битовую скорость передачи данных стандарта, то есть 10 Мбит/с, а слово «Base» — метод передачи на одной базовой частоте[1]

[1] В отличие от методов, использующих несколько несущих частот; такие методы называются широкополосными и имеют в своем составе слово «Broadband». Эти методы хотя и были стандартизованы, не получили распространения в период популярности локальных сетей на разделяемой среде.

(в данном случае — 10 МГц). Последний символ в названии стандарта физического уровня обозначает тип кабеля, в данном случае 5 отражает тот факт, что диаметр «толстого» коаксиала равен 0,5 дюйма. Данный подход к обозначению типа физического уровня Ethernet сохранился до настоящего времени, только вместо диаметра коаксиального кабеля в современных стандартах кодируется тип кабеля (например, 1000Base-T определяет спецификацию для витой пары) или же способ кодирования данных.

В качества метода кодирования сигналов был выбран манчестерский код.

Затем сети Ethernet на «толстом» коаксиальном кабеле были вытеснены сетями на более «тонком» коаксиале (диаметром 0,25 дюйма, что отражает название 10Base-2 этого стандарта), который позволял строить сети более экономичным способом.

Однако сети Ethernet на коаксиальном кабеле обладали одним существенным недостатком, а именно отсутствием оперативной информации о состоянии кабеля и сложностью нахождения места его повреждения. Поэтому поиск неисправностей стал привычной процедурой и головной болью многочисленной армии сетевых администраторов коаксиальных сетей Ethernet.

Альтернатива появилась в середине 80-х годов, когда благодаря использованию витой пары и повторителей сети Ethernet стали гораздо более ремонтопригодными.

К этому времени телефонные компании уже достаточно давно применяли многопарный кабель на основе неэкранированной витой пары для подключения телефонных аппаратов внутри зданий. Идея приспособить этот популярный вид кабеля для локальных сетей оказалась очень плодотворной, так как многие здания уже были оснащены нужной кабельной системой. Оставалось разработать способ подключения сетевых адаптеров и прочего коммуникационного оборудования к витой паре таким образом, чтобы изменения в сетевых адаптерах и программном обеспечении сетевых операционных систем были минимальными по сравнению с сетями Ethernet на коаксиале. Эта попытка оказалась успешной — переход на витую пару требует только замены приемника и передатчика сетевого адаптера, а метод доступа и все протоколы канального уровня остаются теми же, что и в сетях Ethernet на коаксиале. Результатом реализации этой идеи стал стандарт 10Base-T (T — от Twisted pair).

Правда, для соединения узлов в сеть теперь обязательно требуется коммуникационное устройство — **многопортовый повторитель** Ethernet на витой паре.

Устройство такого повторителя схематично изображено на рис. 11.7. Каждый сетевой адаптер соединяется с повторителем двумя витыми парами. Одна витая пара требуется для передачи данных от станции к повторителю (выход T_X сетевого адаптера), другая — для передачи данных от повторителя к станции (вход R_X сетевого адаптера). Повторитель побитно принимает сигналы от одного из конечных узлов и синхронно передает их на все свои остальные порты, исключая тот, с которого поступили сигналы, одновременно улучшая их электрические характеристики.

Многопортовый повторитель часто называют **концентратором**, или **хабом** (от английского hub — центр, ступица колеса), так как в нем сконцентрированы соединения со всеми конечными узлами сети. Фактически хаб имитирует сеть на коаксиальном кабеле в том отношении, что физически отдельные отрезки кабеля на витой паре логически все равно представляют единую разделяемую среду. Все правила доступа к среде по алгоритму CSMA/CD сохраняются.

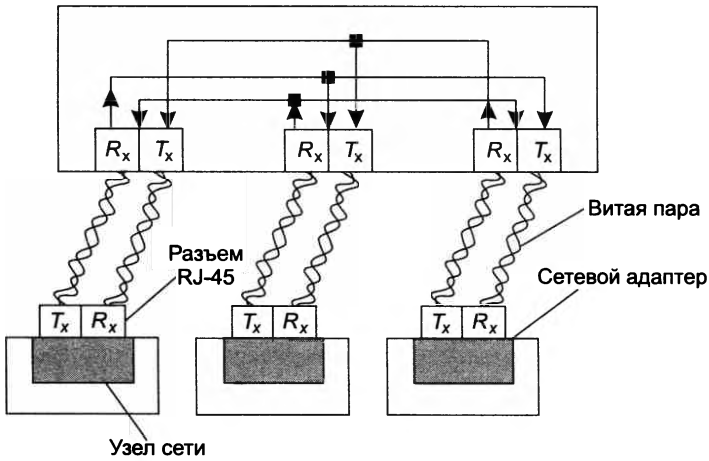


Рис. 11.7. Повторитель Ethernet на витой паре

При создании сети Ethernet на витой паре с большим числом конечных узлов хабы можно соединять друг с другом иерархическим способом, образуя *древовидную структуру* (рис. 11.8). Добавление каждого хаба изменяет физическую структуру, но оставляет без изменения логическую структуру сети. То есть независимо от числа хабов в сети сохраняется одна общая для всех интерфейсов разделяемая среда, так что передача кадра с любого интерфейса блокирует передатчики всех остальных интерфейсов.

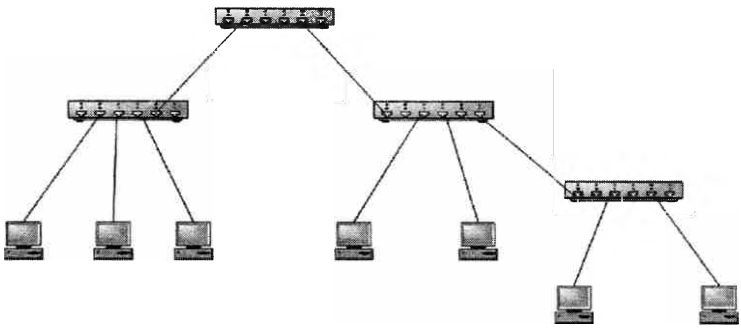


Рис. 11.8. Иерархическое соединение хабов

Физическая структуризация сетей, построенных на основе витой пары, повышает надежность и упрощает обслуживание сети, поскольку в этом случае появляется возможность контролировать состояние и локализовать отказы отдельных кабельных отрезков, подключающих конечные узлы к концентраторам. В случае обрыва, короткого замыкания или неисправности сетевого адаптера работа сети может быть быстро восстановлена путем отключения соответствующего сегмента кабеля.

Для контроля целостности физического соединения между двумя непосредственно соединенными портами в стандарте 10Base-T введен так называемый **тест целостности соединения** (Link Integrity Test, LIT). Эта процедура заключается в том, что в те периоды,

когда порт не посылает или не получает кадры данных, он посылает своему соседу импульсы длительностью 100 нс через каждые 16 мс. Если порт принимает такие импульсы от своего соседа, то он считает соединение работоспособным и, как правило, индицирует это зеленым светом светодиода.

Вскоре появился стандарт 10Base-F, в котором вместо витых пар используется оптическое волокно, обеспечивающее большее расстояние между узлом сети и повторителем.

Во всех вариантах физической среды Ethernet со скоростью 10 Мбит/с используется манчестерский код.

Максимальная производительность сети 10M Ethernet

Производительность сети зависит от скорости передачи кадров по линиям связи и скорости обработки этих кадров коммуникационными устройствами, передающими кадры между своими портами, к которым эти линии связи подключены. Скорость передачи кадров по линиям связи зависит от используемых протоколов физического и канального уровней, например Ethernet на 10 Мбит/с, Ethernet на 100 Мбит/с, Token Ring или FDDI.

Скорость, с которой протокол передает биты по линии связи, называется **номинальной скоростью протокола**.

Скорость обработки кадров коммуникационным устройством зависит от производительности его процессоров, внутренней архитектуры и других параметров. Очевидно, что скорость коммуникационного устройства должна соответствовать скорости работы линии. Если она меньше скорости работы линии, то кадры будут стоять в очередях и отбрасываться при переполнении последних. В то же время нет смысла применять устройство, которое в сотни раз производительнее, чем того требует скорость подключаемых к нему линий.

Для оценки требуемой производительности коммуникационных устройств, имеющих порты Ethernet, необходимо оценить производительность *сегмента Ethernet*, но не в битах в секунду (ее мы знаем — это 10 Мбит/с), а в кадрах в секунду, так как именно этот показатель помогает оценить требования к производительности коммуникационных устройств. Это объясняется тем, что на обработку каждого кадра независимо от его длины мост, коммутатор или маршрутизатор тратят примерно равное время, которое уходит на просмотр таблицы продвижения пакета, формирование нового кадра (для маршрутизатора) и т. п.

При постоянной битовой скорости количество кадров, поступающих на коммуникационное устройство в единицу времени, является, естественно, максимальным при их минимальной длине. Поэтому для коммуникационного оборудования наиболее тяжелым режимом является обработка потока кадров *минимальной длины*.

Теперь рассчитаем максимальную производительность сегмента Ethernet в таких единицах, как число переданных кадров (пакетов) минимальной длины в секунду.

ПРИМЕЧАНИЕ

При указании производительности сетей термины «кадр» и «пакет» обычно используются как синонимы. Соответственно аналогичными являются и единицы измерения производительности: кадры в секунду (кадр/с) и пакеты в секунду (пакет/с).

Для расчета максимального количества кадров минимальной длины, проходящих по сегменту Ethernet, вспомним, что подсчитанное нами ранее время, затрачиваемое на передачу кадра минимальной длины (576 бит), составляет 57,5 мкс. Прибавив межкадровый интервал в 9,6 мкс, получаем, что период следования кадров минимальной длины составляет 67,1 мкс. Отсюда *максимально возможная пропускная способность сегмента Ethernet составляет 14 880 кадр/с* (рис. 11.9). Естественно, что наличие в сегменте нескольких узлов снижает эту величину за счет ожидания доступа к среде, а также из-за коллизий.

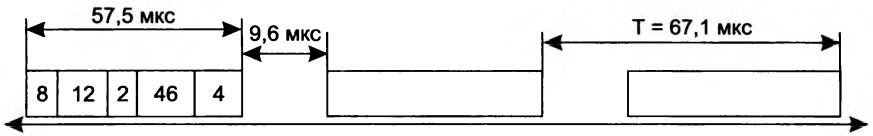


Рис. 11.9. К расчету пропускной способности протокола Ethernet

Кадры максимальной длины технологии Ethernet имеют поле данных в 1500 байт, что вместе со служебной информацией дает 1518 байт, а с преамбулой составляет 1526 байт, или 12 208 бит. *Максимально возможная пропускная способность сегмента Ethernet для кадров максимальной длины составляет 813 кадр/с.* Очевидно, что при работе с большими кадрами нагрузка на мосты, коммутаторы и маршрутизаторы довольно ощутимо снижается.

Теперь рассчитаем, какой максимально полезной пропускной способностью, измеряемой в битах в секунду, обладают сегменты Ethernet при использовании кадров разного размера.

Полезной пропускной способностью протокола называется максимальная скорость передачи *пользовательских* данных, которые переносятся полем данных кадра.

Эта пропускная способность всегда меньше номинальной битовой скорости протокола Ethernet за счет нескольких факторов:

- служебной информации кадра;
- межкадровых интервалов (IPG);
- ожидания доступа к среде.

Для кадров минимальной длины полезная пропускная способность равна:

$$B = 14880 \times 46 \times 8 = 5{,}48 \text{ Мбит/с}.$$

Это несколько меньше, чем 10 Мбит/с, но следует учесть, что кадры минимальной длины используются в основном для передачи квитанций, так что к передаче собственно данных файлов эта скорость имеет небольшое отношение.

Для кадров максимальной длины полезная пропускная способность равна:

$$B_{\text{п}} = 813 \times 1500 \times 8 = 9{,}76 \text{ Мбит/с}.$$

При использовании кадров среднего размера с полем данных в 512 байт пропускная способность протокола составляет 9,29 Мбит/с.

В двух последних случаях пропускная способность протокола оказалась достаточно близкой к предельной пропускной способности в 10 Мбит/с, однако следует учесть, что при

расчете мы предполагали, что двум взаимодействующим станциям «не мешают» никакие другие станции сети, то есть отсутствуют коллизии и ожидание доступа.

Таким образом, при отсутствии коллизий коэффициент использования сети зависит от размера поля данных кадра и имеет максимальное значение 0,976 при передаче кадров максимальной длины.

Важно понимать, что приведенные расчеты пропускной способности справедливы и для коммутируемых сетей Ethernet, так как в них мы не учитывали эффект коллизий. Эти расчеты нетрудно скорректировать и для других битовых скоростей Ethernet, учитывая соответствующий масштабный коэффициент $n \times 10$ (он справедлив и для межкадрового интервала).

Беспроводные локальные сети IEEE 802.11

Проблемы и области применения беспроводных локальных сетей

Беспроводные локальные сети (Wireless Local Area Network, WLAN) в некоторых случаях являются предпочтительным по сравнению с проводными сетями решением, а иногда просто единственно возможным. В WLAN сигнал распространяется с помощью электромагнитных волн высокой частоты. Современные беспроводные локальные сети позволяют передавать данные на скоростях до нескольких гигабит в секунду.

Преимущество беспроводных локальных сетей очевидно — их проще и дешевле разворачивать и модифицировать, так как вся громоздкая кабельная инфраструктура оказывается излишней. Еще одно преимущество — обеспечение мобильности пользователей. Сегодня беспроводные локальные технологии успешно применяются во многих типах сетей: в домашних сетях, в сетях аэропортов, вокзалов, кафе и других публичных местах, для организации временных сетей на различных конференциях и совещаниях, в сетях исторических зданий с уникальной архитектурой, исключающей возможность прокладки кабелей, а также как городские сети, предоставляющие доступ в Интернет на всей территории города.

Однако за эти преимущества беспроводные сети расплачиваются длинным перечнем проблем, которые несет с собой неустойчивая и непредсказуемая беспроводная среда. Мы уже рассматривали особенности распространения сигналов в такой среде в главе 9.

Помехи от разнообразных бытовых приборов и других телекоммуникационных систем, атмосферные помехи и отражения сигнала создают серьезные трудности для надежного приема информации. Локальные сети — это прежде всего сети зданий, а распространение радиосигнала внутри здания еще сложнее, чем вне его. В стандарте IEEE 802.11 приводится изображение распределения интенсивности сигнала (рис. 11.10). В стандарте подчеркивается, что это — статическое изображение, в действительности картина является динамической, и при перемещении объектов в комнате распределение сигнала может существенно измениться.

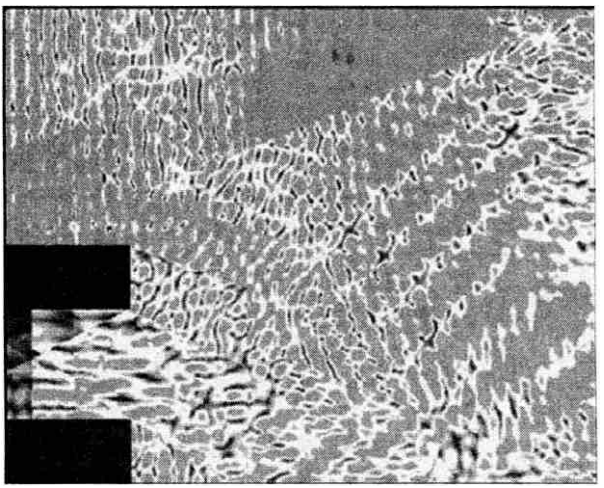


Рис. 11.10. Распределение интенсивности радиосигнала

Методы *расширения спектра* помогают снизить влияние помех на полезный сигнал, кроме того, в беспроводных сетях широко используются *прямая коррекция ошибок* (FEC) и протоколы с повторной передачей потерянных кадров.

Неравномерное распределение интенсивности сигнала приводит не только к битовым ошибкам передаваемой информации, но и к *неопределенности зоны покрытия* беспроводной локальной сети. В проводных локальных сетях такой проблемы нет, те и только те устройства, которые подключены к кабельной системе здания или кампуса, получают сигналы и участвуют в работе LAN. Беспроводная локальная сеть не имеет точной области покрытия. Часто используемое изображение такой области в форме шестиугольника или круга является не чем иным, как абстракцией. В действительности сигнал может быть настолько ослаблен, что устройства, находящиеся в предполагаемых пределах зоны покрытия, вообще не могут принимать и передавать информацию.

Рисунок 11.10 хорошо иллюстрирует такую ситуацию. Подчеркнем, что с течением времени ситуация с распределением сигнала может измениться вместе с изменением состава LAN. По этой причине даже технологии, рассчитанные на фиксированные (не мобильные) узлы сети, должны учитывать то, что беспроводная локальная сеть является неполносвязной. Даже если считать, что сигнал распространяется идеально во все стороны, образованию полносвязной топологии может мешать то, что радиосигнал затухает пропорционально квадрату расстояния от источника. Поэтому при отсутствии базовой станции некоторые пары узлов не смогут взаимодействовать из-за того, что расположены за пределами зоны покрытия передатчиков партнера.

В примере на рис. 11.11, *а* показана такая фрагментированная локальная сеть. Неполносвязность беспроводной сети порождает проблему доступа к разделяемой среде, известную под названием **скрытого терминала**. Проблема возникает в том случае, когда два узла находятся вне зон досягаемости друг друга (узлы *A* и *C* на рис. 11.11, *а*), но существует третий узел *B*, который принимает сигналы как от *A*, так и от *C*. Предположим, что в радиосети используется традиционный метод доступа, основанный на прослушивании несущей, например CSMA/CD. В данном случае коллизии будут возникать значительно чаще, чем

в проводных сетях. Пусть, например, узел *B* занят обменом с узлом *A*. Узлу *C* сложно определить, что среда занята, он может посчитать ее свободной и начать передавать свой кадр. В результате сигналы в районе узла *B* исказятся, то есть произойдет коллизия, вероятность возникновения которой в проводной сети была бы неизмеримо ниже.

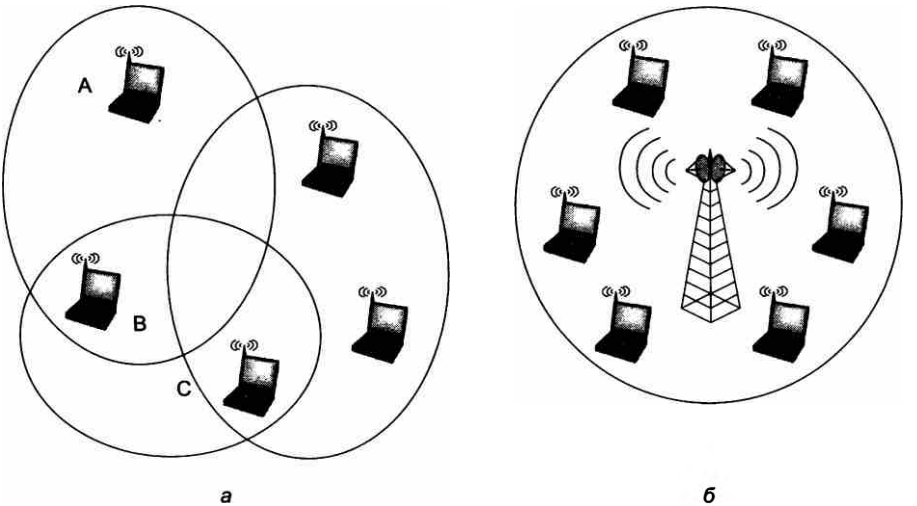


Рис. 11.11. Связность беспроводной локальной сети: *а* — специализированная беспроводная сеть; *б* — беспроводная сеть с базовой станцией

Распознавание коллизий затруднено в радиосети еще и потому, что сигнал собственного передатчика существенно подавляет сигнал удаленного передатчика и распознать искажение сигнала чаще всего невозможно.

> В методах доступа, применяемых в беспроводных сетях, отказываются не только от прослушивания несущей, но и от распознавания коллизий.

Вместо этого в них используют методы предотвращения коллизий, включая **методы опроса**.

Применение базовой станции может улучшить связность сети (рис. 11.11, *б*). Базовая станция обычно обладает большей мощностью, а ее антенна устанавливается так, чтобы более равномерно и беспрепятственно покрывать нужную территорию. В результате все узлы беспроводной локальной сети получают возможность обмениваться данными с базовой станцией, которая транзитом передает данные между узлами.

Далее будет рассмотрен самый популярный стандарт беспроводных локальных сетей — **IEEE 802.11**. Сети и оборудование IEEE.802.11 также известны под названием **Wi-Fi** — по имени консорциума Wi-Fi[1] Alliance, который занимается вопросам совместимости и сертификации оборудования стандартов IEEE 802.11.

---

[1] Wi-Fi является сокращением от Wireless Fidelity — «беспроводная точность»; термин был введен по аналогии с популярным термином Hi-Fi, обозначающим высокую точность воспроизведения звука аппаратурой.

Топологии локальных сетей стандарта 802.11

Стандарт 802.11 поддерживает два типа топологий локальных сетей: с базовым и с расширенным наборами услуг.

Сеть с **базовым набором услуг** (Basic Service Set, BSS) образуется отдельными станциями, базовая станция отсутствует, узлы взаимодействуют друг с другом непосредственно (рис. 11.12). Для того чтобы войти в сеть BSS, станция должна выполнить процедуру присоединения.

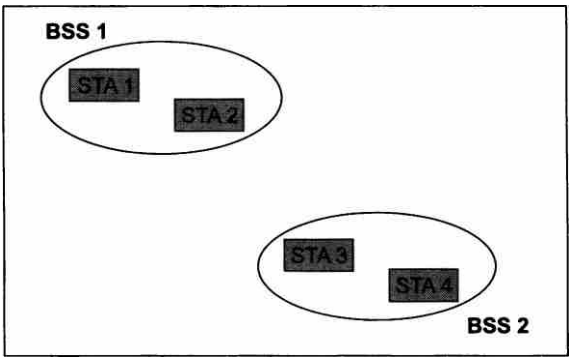


Рис. 11.12. Сети с базовым набором услуг

Сети BSS не являются традиционными сотами в отношении зон покрытия, они могут находиться друг от друга на значительном расстоянии, а могут частично или полностью перекрываться — стандарт 802.11 оставляет здесь свободу для проектировщика сети.

Станции могут использовать разделяемую среду для того, чтобы передавать данные:

- непосредственно друг другу в пределах одной сети BSS;
- в пределах одной сети BSS транзитом через точку доступа;
- между разными сетями BSS через две точки доступа и распределенную систему;
- между сетью BSS и проводной локальной сетью через точку доступа, распределенную систему и портал[1].

В сетях **с расширенным набором услуг** некоторые станции сети являются базовыми, или, в терминологии 802.11, **точками доступа** (Access Point, AP). Станция, которая выполняет функции AP, является членом какой-нибудь сети BSS (рис. 11.13). Все базовые станции сети связаны между собой с помощью распределенной системы (Distribution System, DS), в качестве которой может использоваться та же среда (то есть радио- или инфракрасные волны), что и среда взаимодействия между станциями, или же отличная от нее, например проводная. Точки доступа вместе с распределенной системой поддерживают **службу распределенной системы** (Distribution System Service, DSS). Задачей DSS является передача пакетов между станциями, которые по каким-то причинам не могут или не хотят взаимодействовать между собой непосредственно. Наиболее очевидной причиной использования

[1] Функции портала стандартом не детализируются, это может быть коммутатор или маршрутизатор.

DSS является принадлежность станций разным сетям BSS. В этом случае они передают кадр своей точке доступа, которая через DS передает его точке доступа, обслуживающей сеть BSS со станцией назначения.

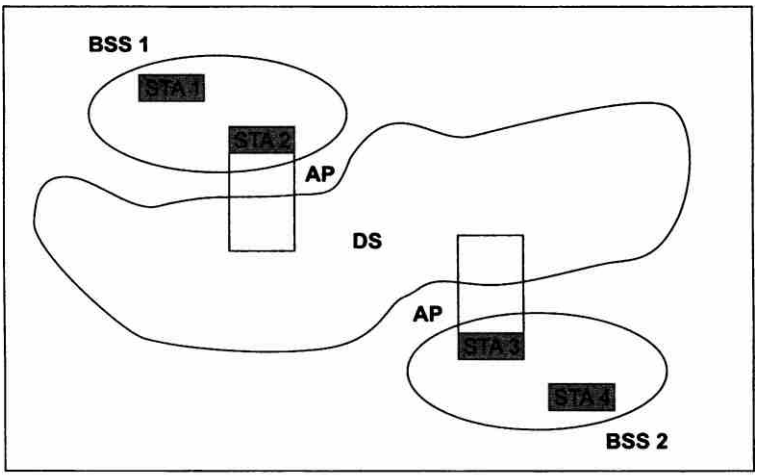


Рис. 11.13. Сеть с расширенным набором услуг

Сеть с **расширенным набором услуг** (Extended Service Set, ESS) состоит из нескольких сетей BSS, объединенных распределенной средой.

Сеть ESS обеспечивает станциям мобильность — они могут переходить из одной сети BSS в другую. Эти перемещения обеспечиваются функциями уровня MAC рабочих и базовых станций, поэтому они совершенно прозрачны для уровня LLC. Сеть ESS может также взаимодействовать с проводной локальной сетью.

Стек протоколов IEEE 802.11

Естественно, что стек протоколов стандарта IEEE 802.11 соответствует общей структуре стандартов комитета 802, то есть состоит из физического уровня и уровня MAC, поверх которых работает уровень LLC. Как и у всех технологий семейства 802, технология 802.11 определяется нижними двумя уровнями, то есть физическим уровнем и уровнем MAC, а уровень LLC выполняет свои стандартные функции, общие для всех технологий LAN.

Структура стека протоколов IEEE 802.11 показана на рис. 11.14.

Уровень MAC выполняет в беспроводных сетях больше функций, чем в проводных. Функции уровня MAC в стандарте 802.11 включают:

- доступ к разделяемой среде;
- обеспечение мобильности станций при наличии нескольких базовых станций;
- обеспечение безопасности, эквивалентной безопасности проводных локальных сетей.

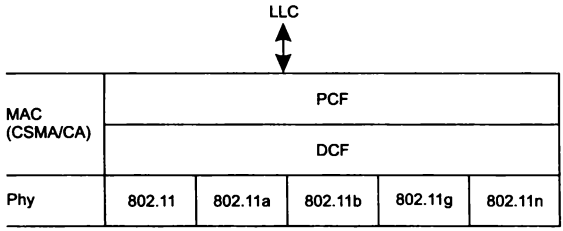


Рис. 11.14. Стек протоколов IEEE 802.11

В сетях 802.11 уровень MAC поддерживает два режима доступа к разделяемой среде: **распределенный режим** (Distributed Coordination Function, **DCF**) и **централизованный режим** (Point Coordination Function, **PCF**). Режим PCF применяется в тех случаях, когда необходимо приоритезировать чувствительный к задержкам трафик.

На физическом уровне существует несколько вариантов спецификаций, которые отличаются используемым частотным диапазоном, методом кодирования и, как следствие, скоростью передачи данных. Все варианты физического уровня работают с одним и тем же алгоритмом уровня MAC, но некоторые временны́е параметры уровня MAC зависят от используемого физического уровня.

Распределенный режим доступа

Рассмотрим сначала, как обеспечивается доступ в распределенном режиме DCF. В этом режиме реализуется метод **CSMA/CA** (Carrier Sense Multiple Access with Collision Avoidance — метод прослушивания несущей частоты с множественным доступом и предотвращением коллизий). Вместо неэффективного в беспроводных сетях прямого распознавания коллизий по методу CSMA/CD здесь они выявляются косвенно. Для этого каждый переданный кадр должен подтверждаться **кадром положительной квитанции**, посылаемым станцией назначения. Если же по истечении оговоренного тайм-аута квитанция не поступает, станция-отправитель считает, что произошла коллизия.

Режим DCF требует синхронизации станций. Она достигается за счет того, что временны́е интервалы начинают отсчитываться от момента окончания передачи очередного кадра (рис. 11.15). Это не требует передачи каких-либо специальных синхронизирующих сигналов.

Станция, которая хочет передать кадр, обязана предварительно прослушать среду. Как только она фиксирует окончание передачи кадра, она обязана отсчитать интервал времени, равный межкадровому интервалу (Inter-Frame Space, IFS). Если после истечения IFS среда все еще свободна, то начинается отсчет слотов фиксированной длительности. Кадр можно начать передавать только в начале какого-либо из слотов при условии, что среда свободна. Станция выбирает для передачи слот на основании *усеченного экспоненциального двоичного алгоритма отсрочки*, аналогичного используемому в методе CSMA/CD. Номер слота выбирается как случайное целое число, равномерно распределенное в интервале [0, CW], где CW означает Contention Window (**конкурентное окно**).

О том, как выбираются размер слота и величина конкурентного окна, рассказывается немного позже, а сейчас рассмотрим этот довольно непростой метод доступа на примере

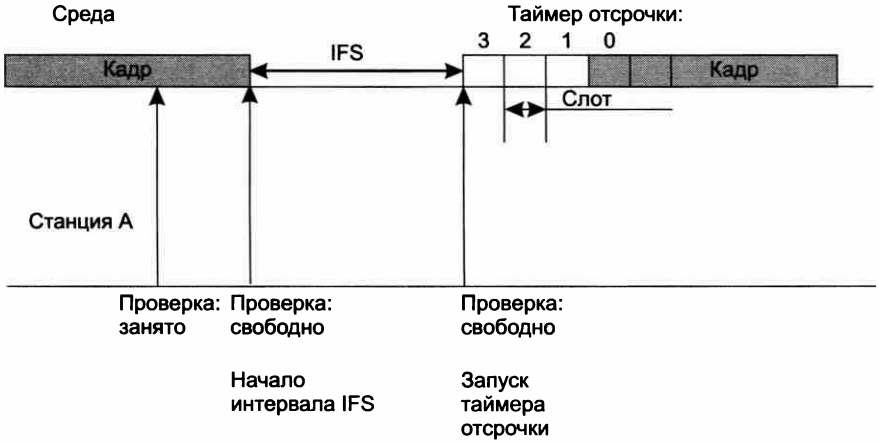


Рис. 11.15. Распределенный режим доступа (DCF)

(см. рис. 11.15). Пусть станция *A* на основании усеченного экспоненциального двоичного алгоритма отсрочки выбрала для передачи слот 3. При этом она присваивает **таймеру отсрочки** (назначение которого будет ясно из дальнейшего описания) значение 3 и начинает проверять состояние среды в начале каждого слота. Если среда свободна, то из значения таймера отсрочки вычитается 1, и если результат равен нулю, то начинается передача кадра.

Таким образом обеспечивается условие незанятости всех слотов, включая выбранный. Это условие является необходимым для начала передачи.

Если же в начале какого-нибудь слота среда оказывается занятой, то вычитания единицы не происходит и таймер «замораживается». В этом случае станция начинает новый цикл доступа к среде. Как и в предыдущем цикле, станция следит за средой и при ее освобождении делает паузу в течение межкадрового интервала. Если среда осталась свободной, то станция *использует значение «замороженного» таймера в качестве номера слота* и выполняет описанную процедуру проверки свободных слотов с вычитанием единиц начиная с замороженного значения таймера отсрочки.

Размер слота выбирается таким образом, чтобы он превосходил время распространения сигнала между любыми двумя станциями сети плюс время, затрачиваемое станцией на распознавание ситуации занятости среды. Размер слота зависит от способа кодирования сигнала. Так, например, для метода кодирования FHSS (см. главу 9) размер слота равен 28 мкс, а для метода DSSS — 1 мкс. Если такое условие соблюдается, то каждая станция сети сумеет правильно распознать начало передачи кадра при прослушивании слотов, предшествующих слоту, выбранному ею для передачи. Это, в свою очередь, означает следующее.

Коллизия может случиться только в том случае, когда несколько станций выбирают один и тот же слот для передачи.

В этом случае кадры искажаются и квитанции подтверждения приема от станций назначения не приходят. Не получив в течение определенного времени квитанцию, отправители фиксируют факт коллизии и пытаются передать свои кадры снова. При каждой повторной неудачной попытке передачи кадра интервал [0, CW], из которого выбирается номер слота, удваивается. Если, например, начальный размер окна выбран равным 8 (то есть CW = 7), то после первой коллизии размер окна должен быть равен 16 (CW = 15), после второй последовательной коллизии — 32 и т. д. Начальное значение CW в соответствии со стандартом 802.11 должно выбираться в зависимости от типа физического уровня, используемого в беспроводной локальной сети.

Как и в методе CSMA/CD, в данном методе количество неудачных попыток передачи одного кадра ограничено, но стандарт 802.11 не дает точного значения этого верхнего предела. Когда верхний предел в *N* попыток достигнут, то кадр отбрасывается, а счетчик последовательных коллизий устанавливается в нуль. Этот счетчик также устанавливается в нуль, если кадр после некоторого количества неудачных попыток все же передается успешно.

В режиме DFC применяются меры для *устранения эффекта скрытого терминала*. Для этого станция, которая хочет захватить среду и в соответствии с описанным алгоритмом начинает передачу кадра в определенном слоте, вместо кадра данных сначала посылает станции назначения короткий служебный кадр RTS (Request To Send — запрос на передачу). На этот запрос станция назначения должна ответить служебным кадром CTS (Clear To Send — свободна для передачи), после чего станция-отправитель посылает кадр данных. Кадр CTS должен оповестить о захвате среды те станции, которые находятся вне зоны сигнала станции-отправителя, но в зоне досягаемости станции-получателя, то есть являются скрытыми терминалами для станции-отправителя.

Централизованный режим доступа

В том случае, когда в сети BSS имеется станция, выполняющая функции точки доступа, может применяться также централизованный режим доступа (PCF), обеспечивающий приоритетное обслуживание трафика. В этом случае говорят, что точка доступа играет роль арбитра среды.

Режим PCF в сетях 802.11 сосуществует с режимом DCF. Оба режима координируются с помощью трех типов межкадровых интервалов (рис. 11.16).

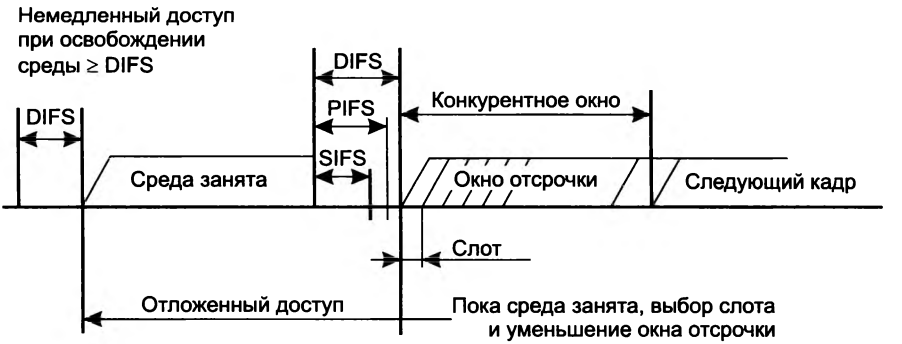


Рис. 11.16. Сосуществование режимов PCF и DCF

После освобождения среды каждая станция отсчитывает время простоя среды, сравнивая его с тремя значениями:

- короткий межкадровый интервал (Short IFS, **SIFS**);
- межкадровый интервал режима PCF (**PIFS**);
- межкадровый интервал режима DCF (**DIFS**).

Захват среды с помощью распределенной процедуры режима DCF возможен только в том случае, когда среда свободна в течение времени, равного или большего, чем DIFS. То есть в качестве IFS в режиме DCF нужно использовать интервал DIFS — самый длительный период из трех возможных, что дает этому режиму самый низкий приоритет.

Межкадровый интервал SIFS имеет наименьшее значение, он служит для первоочередного захвата среды ответными кадрами CTS или квитанциями, которые продолжают или завершают уже начавшуюся передачу кадра.

Значение межкадрового интервала PIFS больше, чем SIFS, но меньше, чем DIFS. Промежутком времени между завершением PIFS и DIFS пользуется арбитр среды. В этом промежутке он может передать специальный кадр, который говорит всем станциям, что начинается **контролируемый период**. Получив этот кадр, станции, которые хотели бы воспользоваться алгоритмом DCF для захвата среды, уже не могут этого сделать, они должны дожидаться окончания контролируемого периода. Длительность этого периода объявляется в специальном кадре, но этот период может закончиться и раньше, если у станций нет чувствительного к задержкам трафика. В этом случае арбитр передает служебный кадр, после которого по истечении интервала DIFS начинает работать режим DCF.

На управляемом интервале реализуется *централизованный метод доступа* (PCF). Арбитр выполняет процедуру опроса, чтобы по очереди предоставить каждой такой станции право на использование среды, направляя ей специальный кадр. Станция, получив такой кадр, может ответить другим кадром, который подтверждает прием специального кадра и одновременно передает данные (либо по адресу арбитра для транзитной передачи, либо непосредственно станции).

Для того чтобы какая-то доля среды всегда доставалась асинхронному трафику, длительность контролируемого периода ограничена. После его окончания арбитр передает соответствующий кадр и начинается неконтролируемый период.

Каждая станция может работать в режиме PCF, для этого она должна подписаться на эту услугу при присоединении к сети.

Физические уровни стандарта 802.11

С момента принятия первой версии стандарта 802.11 в 1997 году одной из главных проблем, над которой работали специалисты, занимающиеся развитием беспроводных локальных сетей, была проблема повышения скорости передачи данных, чтобы приложения, хорошо работающие в проводных сетях, при переходе на беспроводную связь значительно не деградировали.

Другой немаловажной проблемой был выбранный диапазон частот радиоспектра. В соответствии с рекомендациями ITU диапазоны 2,4, 3,6 и 5 ГГц отведены для беспроводной передачи данных, при этом лицензирование этих диапазонов не рекомендуется. В разных странах существуют различные правила выбора этих диапазонов (причем правила для

каждого из диапазонов могут быть разными), от свободного использования до обычного лицензирования. Помимо беспроводных локальных сетей в этих диапазонах могут работать и другие типы устройств, например любительское радио или беспроводные сети городов.

Физические уровни стандарта 802.11 1997 года

В 1997 году комитетом **802.11** был принят стандарт, который определял функции уровня MAC вместе с *тремя вариантами физического уровня*, которые обеспечивают передачу данных со скоростями 1 и 2 Мбит/с.

- В первом варианте средой являются *инфракрасные волны* диапазона 850 нм, которые генерируются либо полупроводниковым лазерным диодом, либо светодиодом (LED). Так как инфракрасные волны не проникают через стены, область покрытия LAN ограничивается зоной прямой видимости. Стандарт предусматривает три варианта распространения излучения: ненаправленную антенну, отражение от потолка и фокусное направленное излучение. В первом случае узкий луч рассеивается с помощью системы линз. Фокусное направленное излучение предназначено для организации двухточечной связи, например между двумя зданиями.
- Во втором варианте в качестве передающей среды используется *микроволновый диапазон* 2,4 ГГц. Этот вариант основан на методе FHSS. В методе FHSS каждый узкий канал имеет ширину 1 МГц. Частотная манипуляция (FSK) с двумя состояниями сигнала (частотами) дает скорость 1 Мбит/с, с четырьмя состояниями — 2 Мбит/с. В случае FHSS сеть может состоять из сот, причем для исключения взаимного влияния в соседних сотах могут применяться ортогональные последовательности частот. Количество каналов и частота переключения между каналами настраиваются, так что при развертывании беспроводной локальной сети можно учитывать особенности регулирования спектра частот конкретной страны.
- Третий вариант, в котором используется тот же *микроволновый диапазон*, основан на методе DSSS, где в качестве последовательности чипов применяется 11-битный код 10110111000. Каждый бит кодируется путем двоичной фазовой (1 Мбит/с) или квадратурной фазовой (2 Мбит/с) манипуляции.

Физические уровни стандартов 802.11a и 802.11b

В 1999 году были приняты два варианта стандарта физического уровня: **802.11a** и **802.11b**, заменяющие спецификации физического уровня 802.11 редакции 1997 года.

В *спецификации 802.11b института IEEE* по-прежнему используется диапазон 2,4 ГГц. Для повышения скорости до 11 Мбит/с, которая сопоставима со скоростью классического стандарта Ethernet, здесь применяется более эффективный вариант метода DSSS, опирающийся на технику Complementary Code Keying (CCK), заменившую коды Баркера.

Однако диапазон 2,4 ГГц с шириной полосы примерно в 80 МГц используется стандартом 802.11b, отличным от стандарта 1997 года способом. Этот диапазон разбит на 14 каналов, каждый из которых, кроме последнего, отстоит от соседей на 5 МГц (рис. 11.17).

Для передачи данных согласно стандарту 802.11b используется полоса частот шириной в 22 МГц, поэтому одного канала шириной в 5 МГц оказывается недостаточно, приходится объединять несколько соседних каналов. Для того чтобы гарантировать некоторый минимум взаимных помех, возникающих от передатчиков, работающих в диапазоне 2,4 ГГц,

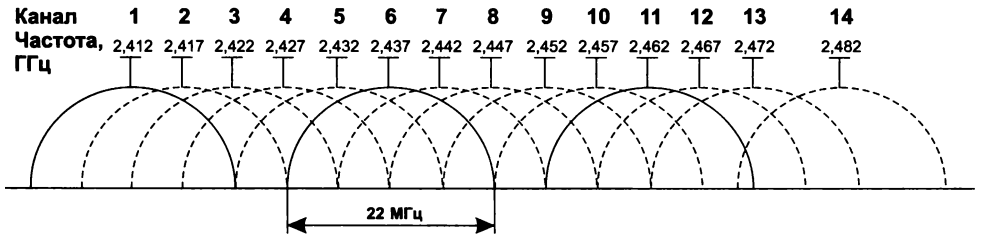


Рис. 11.17. Разбиение диапазона 2,4 ГГц на каналы

комитет 802.11 определил так называемую спектральную маску, определяющую разрешенный спектр мощности передатчика, работающего в каком-либо из каналов. Этот спектр должен затухать не меньше чем на 30 дБ на расстоянии 11 МГц от центра канала, что и создает укрупненную полосу шириной в 22 МГц с центром в некотором из 14 каналов.

В результате одновременно в одной и той же области покрытия могут работать несколько независимых беспроводных сетей стандарта 802.11b. На рис. 11.17 показан вариант для трех сетей, использующих каналы 1, 6 и 11. Такое использование каналов типично для США, где частотные каналы 12, 13 и 14 для сетей стандарта 802.11 не разрешены. В Европе в конце 90-х годов действовали более жесткие ограничения, например в Испании были разрешены только каналы 10 и 11, а во Франции — только каналы 10, 11, 12 и 13, но постепенно эти ограничения были сняты, и сейчас лишь канал 14 в большинстве стран по-прежнему не задействован. Таким образом, в странах Европы максимальное количество независимых сетей, работающих в одной области покрытия, достигает четырех; обычно они занимают каналы 1, 5, 9 и 13.

Оборудование стандарта 802.11b может конфигурироваться для любого из 14 каналов диапазона 2,4 ГГц, так что при возникновении помех на определенном канале можно перейти на другой.

Спецификация 802.11a обеспечивает повышение скорости передачи данных за счет использования полосы частот шириной 300 МГц из диапазона частот 5 ГГц. Так как полоса частот, отведенная для беспроводных локальных сетей, в этом диапазоне шире, то и количество каналов шириной в 5 МГц здесь больше, чем в диапазоне 2,4 ГГц, — в зависимости от правил регулирования конкретной страны их может быть 48 и более. Для передачи данных в технологии задействована полоса частот шириной 20 Мгц, что дает возможность иметь 12 и более независимых сетей в одной области покрытия.

Для передачи данных в стандарте 802.11a используется техника ортогонального частотного мультиплексирования (OFDM). Данные первоначально кодируются на 52 первичных несущих частотах методом BPSK, QPSK, 16-QAM или 64-QAM, а затем сворачиваются в общий сигнал с шириной спектра в 20 МГц. Скорость передачи данных в зависимости от метода кодирования первичной несущей частоты составляет 6, 9, 12, 18, 24, 36, 48 или 54 Мбит/с.

Диапазон 5 ГГц в спецификации 802.11a пока меньше «населен» и предоставляет больше частотных каналов для передачи данных. Однако его использование связано с несколькими проблемами. Во-первых, оборудование для этих частот пока еще слишком дорогое, во-вторых, в некоторых странах частоты этого диапазона подлежат лицензированию, в-третьих, волны этого диапазона хуже проходят через препятствия.

Физический уровень стандарта 802.11g

Стандарт **802.11g** для физического уровня разработан рабочей группой института IEEE летом 2003 года. Он быстро завоевал популярность, так как обеспечивал те же скорости, что и стандарт 802.11a, то есть до 54 Мбит/с, но в диапазоне 2,4 ГГц, то есть в том диапазоне, где до этого удавалось достигать максимальной скорости в 11 Мбит/с на оборудовании стандарта 802.11b. В то же время стоимость оборудования стандарта 802.11g достаточно быстро стала соизмеримой со стоимостью оборудования стандарта 802.11b, что и стало причиной роста популярности новой спецификации. В ней, так же как и в спецификации 802.11a, используется ортогональное частотное мультиплексирование (OFDM).

Диаметр сети стандарта 802.11g зависит от многих параметров, в том числе от используемого диапазона частот. Обычно диаметр беспроводной локальной сети находится в пределах от 100 до 300 м вне помещений и от 30 до 40 м внутри помещений.

Физический уровень стандарта 802.11n

Стандарт 802.1n был принят в октябре 2009 года. Основной его особенностью является дальнейшее повышение скорости передачи данных (до 600 Мбит/с). Оборудование стандарта 802.11n может работать как в диапазоне 5 ГГц, так и в диапазоне 2,4 ГГц, хотя рекомендуемым диапазоном является диапазон 5 ГГц благодаря большему числу доступных каналов и меньшей интерференции с многочисленным оборудованием, работающим сегодня в диапазоне 2,4 ГГц.

Для достижения высоких скоростей в технологии 802.11n применено несколько новых механизмов.

- *Улучшенное кодирование OFDM и сдвоенные частотные каналы.* Вместо каналов с полосой в 20 МГц, которые использовались в технологиях 802.11a и 802.11g, в технологии 802.11n применены каналы с полосой 40 МГц (для обратной совместимости допускается также работать с каналами 20 МГц). Само по себе расширение полосы в два раза должно приводить к повышению битовой скорости в два раза, но выигрыш здесь больше за счет усовершенствований в кодировании OFDM: вместо 52 первичных несущих частот на полосу в 20 МГц здесь используется 57 таких частот, а на полосу в 40 Мгц соответственно 114. Это приводит к повышению битовой скорости с 54 до 65 Мбит/с для каналов 20 Мгц и до 135 Мбит/с для каналов 40 МГц.
- *Уменьшение межсимвольного интервала.* Для надежного распознавания кодовых символов в технологиях 80.11a/g используется межсимвольный интервал в 800 нс. Технология 802.11n позволяет передавать данные с таким же межсимвольным интервалом, а также с межсимвольным интервалом в 400 нс, что повышает битовую скорость для каналов 40 МГц до 150 Мбит/с.
- *Применение техники MIMO* (Multiple Input Multiple Output — множественные входы и выходы). Эта техника основана на использовании одним сетевым адаптером нескольких антенн с целью лучшего распознавания сигнала, пришедшего к приемнику разными путями. Обычно из-за таких эффектов распространения радиоволн, как отражение, дифракция и рассеивание, приемник получает несколько сигналов, дошедших от передатчика по разным физическим путям и имеющим, следовательно, сдвиг по

фазе. До введения техники MIMO такие явления считались негативными и с ними боролись путем применения нескольких (обычно двух) антенн, из которых в каждый момент времени использовалась только одна — та, которая принимала сигнал лучшего качества. Техника MIMO принципиально изменила отношение к сигналам, пришедшим разными путями, — эти сигналы комбинируются и путем цифровой обработки из них восстанавливается исходный сигнал.

Техника MIMO не только способствует улучшению соотношения сигнал/помеха. Благодаря возможности обрабатывать сигналы, пришедшие разными путями, для каждого потока с целью создания избыточного сигнала можно передавать с помощью нескольких антенн несколько независимых потоков данных (обычно их число меньше, чем число антенн). Эта способность систем MIMO называется **пространственным мультиплексированием** (spatial multiplexing). Для систем MIMO принято использовать обозначение:

$$T \times R : S.$$

Здесь T — количество передающих антенн узла, R — количество принимающих антенн узла, а S — количество потоков данных, которые пространственно мультиплексируются. Типичный системой MIMO стандарта 802.11n является система $3 \times 3 : 2$, то есть система с тремя передающими и тремя принимающими антеннами, которая позволяет передавать два независимых потока данных. Система MIMO $3 \times 3 : 2$ обеспечивает повышение битовой скорости в два раза, то есть до 300 Мбит/с для каналов 40 МГц. Стандарт 802.11n предусматривает различные варианты системы MIMO вплоть до $4 \times 4 : 4$, что позволяет повысить битовую скорость до 600 Мбит/с.

Физический уровень стандарта 802.11ac

Спецификация 802.11ac является развитием спецификации 802.11n, она обеспечивает скорости передачи данных до 1 Гбит/с за счет:

- расширения полосы индивидуального канала до 80 МГц (обязательная опция) или 160 МГц (возможная опция);
- поддержки до 8 каналов MIMO;
- применения модуляции сигнала с большим числом состояний: 256QAM вместо 64QAM у стандарта 802.11n.

Физический уровень стандарта 802.11ad

Эта версия стандарта 802.11 была создана беспроводным гигабитным альянсом (Wireless Gigabit Alliance, WiGig) в 2009–2011 годах. Стандарт отличается тем, что в нем используется частотный диапазон 60 ГГц (а также диапазоны 2,4 и 5 ГГц). В диапазоне 60 ГГц может существовать четыре канала шириной 2,16 ГГц каждый. Широкая полоса канала позволяет передавать данные с гигабитными скоростями — до 4,6 Гбит/с при наличии одного канала и до 7 Гбит/с при мультиплексированой передаче OFDM одновременно по четырем каналам.

Однако передача данных с несущей частотой 60 ГГц сталкивается с проблемой распространения сигнала — он не проходит через стены, как сигнал частот 2,4 или 5 ГГц (хотя и отражается от препятствий, что используют приемопередатчики 802.11ad). Поэтому область покрытия сети 802.11ad ограничена одной комнатой, в которой находятся устройства,

требующие обмена данными с гигабитными скоростями, например приемник и телевизор стандарта HD.

Приемопередатчики стандарта 802.11ad могут также работать в диапазонах 2,4 и 5 ГГц, чтобы обеспечить связь, когда невозможно использовать сигнал частоты 60 ГГц (из-за размещения узлов сети в разных помещениях или из-за слишком большого расстояния).

Персональные сети и технология Bluetooth

Особенности персональных сетей

Персональные сети (Personal Area Network, PAN) предназначены для взаимодействия устройств, принадлежащих одному владельцу, на небольшом расстоянии, обычно *в радиусе 10 м*. Такими устройствами могут быть ноутбук, мобильный телефон, принтер, карманный компьютер (Personal Digital Assistant, PDA), телевизор, а также многочисленные бытовые приборы, например холодильник.

Персональные сети связывают устройства, принадлежащие, как правило, одному пользователю, на небольших расстояниях. Типичным примером PAN является беспроводное соединение компьютера с периферийными устройствами, такими как принтер, наушники, мышь, клавиатура и т. п. Мобильные телефоны также используют технологию PAN для соединения со своей периферией (чаще всего это наушники), а также с компьютером своего владельца. Некоторые марки наручных часов стали поддерживать технологию PAN, превращаясь в универсальные устройства с функциями PDA.

Персональные сети должны обеспечивать как фиксированный доступ, например в пределах дома, так и мобильный, когда владелец устройств PAN перемещается вместе с ними между помещениями или городами.

Персональные сети во многом похожи на локальные, но у них есть и свои особенности.

- Многие из устройств, которые могут входить в персональную сеть, *гораздо проще*, чем традиционный узел LAN — компьютер. Кроме того, такие устройства обычно имеют небольшие габариты и стоимость. Поэтому в стандартах PAN требуется учитывать, что их реализация должна приводить к недорогим решениям с низким энергопотреблением.
- *Область покрытия PAN меньше области покрытия LAN*, узлы PAN часто находятся на расстоянии нескольких метров друг от друга.
- *Высокие требования к безопасности.* Персональные устройства, путешествуя вместе со своим владельцем, попадают в различное окружение. Иногда они должны взаимодействовать с устройствами других персональных сетей, например если их владелец встретил на улице своего знакомого и решил переписать из его устройства PDA в свое несколько адресов общих знакомых. В других случаях такое взаимодействие явно нежелательно, так как может привести к утечке конфиденциальной информации. Поэтому протоколы PAN должны обеспечивать разнообразные методы аутентификации устройств и шифрования данных в мобильной обстановке.
- При соединении малогабаритных устройств между собой желание избавиться от кабелей проявляется гораздо сильнее, чем при соединении компьютера с принтером

или концентратором. Из-за этого персональные сети в гораздо большей степени, чем локальные, *тяготеют к беспроводным решениям*.

- Если человек постоянно носит устройство PAN с собой и на себе, то оно не должно причинять вреда его здоровью. Поэтому такое устройство должно *излучать сигналы небольшой мощности*, желательно не более 100 мВт (обычный сотовый телефон излучает сигналы мощностью от 600 мВт до 3 Вт).

Сегодня самой популярной технологией PAN является **Bluetooth**, которая обеспечивает взаимодействие восьми устройств в разделяемой среде диапазона 2,4 МГц с битовой скоростью передачи данных до 3 Мбит/с.

Архитектура Bluetooth

Стандарт Bluetooth разработан группой Bluetooth SIG (Bluetooth Special Interest Group), которая была организована по инициативе компании Ericsson. Стандарт Bluetooth также адаптирован рабочей группой IEEE 802.15.1 в соответствии с общей структурой стандартов IEEE 802.

В технологии Bluetooth используется концепция **пикосети**. Название подчеркивает небольшую область покрытия, от 10 до 100 м, в зависимости от мощности излучения передатчика устройства. В пикосеть может входить до 255 устройств, но только восемь из них могут в каждый момент времени быть активными и обмениваться данными. Одно из устройств в пикосети является **главным**, остальные — **подчиненными** (рис. 11.18).

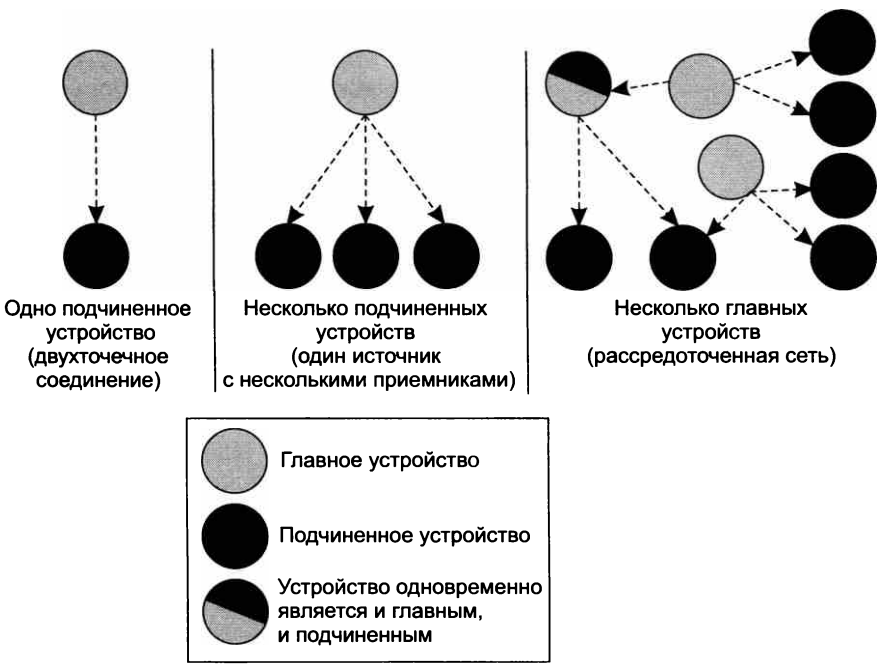


Рис. 11.18. Пикосеть и рассредоточенная сеть

Активное подчиненное устройство может обмениваться данными только с главным устройством, прямой обмен между подчиненными устройствами невозможен. Все подчиненные устройства данной пикосети, кроме семи активных, должны находиться в режиме пониженного энергопотребления, в котором они только периодически прослушивают команду главного устройства для перехода в активное состояние.

Главное устройство отвечает за доступ к *разделяемой среде пикосети*, которая представляет собой нелицензируемые частоты диапазона 2,4 ГГц. Разделяемая среда передает данные со скоростью до 3 Мбит/с, но из-за накладных расходов на заголовки пакетов и смену частот полезная скорость передачи данных в среде не превышает 2,1 Мбит/с. Пропускная способность среды делится главным устройством между семью подчиненными устройствами на основе техники TDM.

Такая архитектура позволяет применять более простые протоколы в устройствах, выполняющих функции подчиненных (например, в радионаушниках), и отдает более сложные функции управления пикосетью компьютеру, который, скорее всего, и будет главным устройством этой сети.

Присоединение к пикосети происходит динамически. Главное устройство пикосети, используя процедуру опроса, собирает информацию об устройствах, которые попадают в зону его пикосети. После обнаружения нового устройства главное устройство проводит с ним переговоры. Если желание подчиненного устройства присоединиться к пикосети совпадает с решением главного устройства (подчиненное устройство прошло проверку аутентичности и оказалось в списке разрешенных устройств), то новое подчиненное устройство присоединяется к сети.

ПРИМЕЧАНИЕ

Безопасность сетей Bluetooth обеспечивается за счет аутентификации устройств и шифрования передаваемого трафика. Протоколы Bluetooth предлагают более высокий уровень защиты, чем протокол WEP стандарта IEEE 802.11.

Несколько пикосетей, которые обмениваются между собой данными, образуют **рассредоточенную сеть**. Взаимодействие в пределах рассредоточенной сети осуществляется за счет того, что один узел (называемый *мостом*) одновременно является членом нескольких пикосетей, причем этот узел может исполнять роль главного устройства одной пикосети и подчиненного устройства другой.

Сеть Bluetooth использует технику расширения спектра FHSS. Для того чтобы сигналы разных пикосетей не интерферировали, каждое главное устройство задействует *собственную* последовательность псевдослучайной перестройки частоты. Наличие различающихся последовательностей псевдослучайной перестройки частоты затрудняет общение пикосетей между собой. Для преодоления этой проблемы устройство, играющее роль моста, должно при подключении к каждой из пикосетей соответствующим образом менять последовательность.

Коллизии, хотя и с очень небольшой вероятностью, все же могут происходить, когда два или более устройства из разных пикосетей выберут для работы один и тот же частотный канал.

Для надежной передачи данных в технологии Bluetooth может выполняться прямая коррекция ошибок (FEC), а получение кадра подтверждается с помощью квитанций.

В сетях Bluetooth для передачи информации двух типов используются разные методы.

- Для *чувствительного к задержкам трафика* (например, голоса) сеть поддерживает **синхронный канал, ориентированный на соединение** (Synchronous Connection-Oriented link, SCO). Этот канал работает на скорости 64 Кбит/с. Для канала SCO пропускная способность резервируется на все время соединения.
- Для *эластичного трафика* (например, компьютерных данных) используется работающий с переменной скоростью **асинхронный канал, не ориентированный на соединение** (Asynchronous Connection-Less link, ACL). Для канала ACL пропускная способность выделяется по запросу подчиненного устройства или по потребности главного устройства.

Bluetooth является законченной оригинальной технологией, рассчитанной на самостоятельное применение в электронных персональных устройствах. Поэтому эта технология поддерживает полный стек протоколов, включая собственные прикладные протоколы. В этом заключается ее отличие от рассмотренных ранее технологий, таких как Ethernet или IEEE 802.11, которые лишь выполняют функции физического и канального уровней.

Создание для технологии Bluetooth собственных прикладных протоколов объясняется стремлением разработчиков реализовывать ее в разнообразных простых устройствах, которым не под силу, да и ни к чему поддерживать стек протоколов TCP/IP. Кстати, технология Bluetooth появилась в результате попыток разработать стандарт для взаимодействия мобильного телефона с беспроводными наушниками. Понятно, что для решения такой простой задачи не нужен ни протокол передачи файлов (FTP), ни протокол передачи гипертекста (HTTP). В результате для технологии Bluetooth был создан оригинальный стек протоколов, в дополнение к которому появилось большое количество профилей.

Стек протоколов Bluetooth постоянно совершенствуется. Версия 1.0 стандартов стека была принята в 1999 году, версия 1.2 — в 2003, версия 2.0 — в 2004, версия 2.1 — в 2007, версия 3.0 — в 2009, версия 4.0 — в 2010, версия 4.1 — в 2013, а версия 4.2 — в декабре 2014 года.

Профили определяют конкретный набор протоколов для решения той или иной задачи. Например, существует профиль для взаимодействия компьютера или мобильного телефона с беспроводными наушниками. Имеется также профиль для тех устройств, которые могут передавать файлы (наушникам он, скорее всего, не потребуется, хотя будущее предвидеть сложно), профиль эмуляции последовательного порта RS-232 и т. д.

Разделяемая среда представляет собой последовательность частотных каналов технологии FHSS в диапазоне 2,4 ГГц. Каждый частотный канал имеет ширину 1 МГц, количество каналов равно 79 (в США и большинстве других стран мира) или 23 (в Испании, Франции, Японии).

Чиповая скорость равна 1600 Гц, поэтому период чипа составляет 625 мкс. Главное устройство разделяет общую среду на основе временно́го мультиплексирования (TDM), используя в качестве тайм-слота время пребывания системы на одном частотном канале, то есть 625 мкс. В версии протоколов 1.0 информация кодируется с тактовой частотой 1 МГц путем двоичной частотной манипуляции (BFSK), в результате битовая скорость составляет 1 Мбит/с. В течение одного тайм-слота пикосеть Bluetooth передает 625 бит, но не все они служат для

передачи полезной информации. При смене частоты устройствам сети требуется некоторое время для синхронизации, поэтому из 625 бит только 366 передают кадр данных.

В версии 2.0 был введен режим **улучшенной скорости передачи данных** (Enhanced Data Rate, EDR), в котором для кодирования данных используется комбинация методов частотной (BFSK) и фазовой (PSK) модуляции; за счет этого удалось повысить битовую скорость до 3 Мбит/с, а полезную скорость передачи данных — до 2,1 Мбит/с. Режим EDR дополняет основной режим передачи данных со скоростью 1 Мбит/с.

Кадр данных может занимать 1, 3 или 5 слотов. Когда кадр занимает больше одного слота, частота канала остается неизменной в течение всего времени передачи кадра. В этом случае накладные расходы на синхронизацию меньше, так что размер кадра, состоящего, например, из пяти последовательных слотов, равен 2870 бит (с полем данных до 2744 бит).

Поиск и стыковка устройств Bluetooth

Устройство, поддерживающее технологию Bluetooth, обычно посылает периодические запросы на предмет обнаружения других устройств Bluetooth в зоне досягаемости. Если устройство Bluetooth получает такой запрос и оно сконфигурировано таким образом, чтобы отвечать на запросы, то в ответ устройство передает сведения о себе: имя и тип устройства, имя производителя, поддерживаемые сервисы.

Имя устройства конфигурируется в отличие от его уникального MAC-адреса, который дается производителем. Нужно отметить, что часто устройства выпускаются со сконфигурированными по умолчанию именами, соответствующими названию модели устройства, поэтому в сфере досягаемости вашего мобильного телефона может оказаться несколько других телефонов с одинаковыми именами Bluetooth, если их владельцы не дали им собственные имена.

После предварительного обмена информацией устройства Bluetooth могут начать так называемую процедуру стыковки (pairing), если конфигурация устройств ее требует. Стыковка подразумевает установление защищенного канала (см. главу 29) между устройствами; безопасность в данном случае означает, что устройства доверяют друг другу, а данные между ними передаются в зашифрованном виде. Стыковка устройств Bluetooth требует введения в каждое из них одного и того же пароля, называемого также PIN-кодом Bluetooth. Обычно устройство, получившее запрос на стыковку, просит пользователя ввести PIN-код. Устройства, успешно прошедшие процедуру стыковки, запоминают этот факт и устанавливают безопасное соединение автоматически всякий раз, когда оказываются в зоне досягаемости, при этом повторное введение PIN-кода пользователем не требуется.

Устройство сможет быть сконфигурировано пользователем или производителем таким образом, чтобы разрешать установление соединений с другими устройствами без процедуры стыковки.

Развитие технологии Bluetooth

В последних версиях стандартов Bluetooth были анонсированы некоторые нововведения, одно из которых — повышение скорости передачи данных в режиме EDR до 3 Мбит/с — мы уже упомянули. Далее перечислены другие наиболее важные новые свойства этой технологии.

- *Пониженная скорость обмена в ждущем режиме.* Это свойство заключается в снижении частоты обмена служебными сообщениями keepalive («работоспособен»), которыми узлы поддерживают соединение в открытом состоянии при отсутствии пользовательских данных для передачи, с нескольких сообщений в секунду до одного сообщения раз в 5 или 10 секунд. Такой режим позволяет увеличить время работы батарей портативных устройств в 3–10 раз. Свойство введено в версии 2.1.
- *Безопасная простая стыковка* (secure simple pairing) позволяет ускорить процедуру стыковки и в то же время предлагает более высокую степень защиты соединений. Свойство введено в версии 2.1.
- *Использование технологии NFC* (Near Field Communication — связь ближнего радиуса действия) для автоматической стыковки устройств. NFC — это новая технология, разработанная для беспроводного взаимодействия устройств на расстояниях в 10–20 см. При обнаружении сигналов устройства с интерфейсами NFC автоматически устанавливают соединение. Устройства Bluetooth могут использовать технологию NFC для автоматического обнаружения при приближении их друг к другу в ходе стыковки и обмена информацией. Это свойство является частью упомянутой ранее процедуры безопасной простой стыковки, оно также введено в версии 2.1 Bluetooth.
- *Альтернативные MAC-уровень и физический уровень.* При необходимости передачи большого объема данных устройство Bluetooth может переключиться на соединение, использующее отличную от Bluetooth технологию передачи данных. В версии 3.0 протоколов Bluetooth как возможная альтернатива определены пока только технологии 802.11, но в будущем могут быть стандартизованы и другие технологии. Первоначальное взаимодействие устройств всегда должно производиться на основе технологии Bluetooth.
- *Bluetooth с низким энергопотреблением* (Bluetooth low energy). В апреле 2009 года группа Bluetooth SIG объявила о совершенно новом дополнительном стеке протоколов под названием Bluetooth Low Energy (Bluetooth LE). Этот стек разрабатывался группой Bluetooth SIG совместно с компанией Nokia и был первоначально известен под названием Wibree. Протоколы Bluetooth LE предназначены для устройств, батареи которых должны иметь примерно годичный срок действия; это могут быть, например, наручные часы или медицинские приборы.

  Технология Bluetooth LE получила маркетинговое название Bluetooth Smart, сегодня она реализована в большинстве смарт-телефонов и планшетов. Существуют реализации протоколов Bluetooth Smart в качестве единственного стека протоколов Bluetooth некоторого устройства, а также в варианте второго стека протоколов, работающего наряду с классическим стеком, — в этом случае такое устройство маркируется как Bluetooth Smart Ready.

  Для передачи данных Bluetooth Smart использует тот же частотный диапазон 2,4 ГГц, что и классический вариант Bluetooth, но в нем организуется не 79 каналов с полосой 1 МГц, а 40 каналов с полосой 2 МГц каждый. Битовая скорость передачи данных составляет 1 Мбит/с, то есть такая же, как и у классической версии. Передача голоса по Bluetooth Smart не предусмотрена.

  Спецификация Bluetooth Smart описана в версии Bluetooth SIG 4.0 стандарта, а в версии 4.2 вводятся некоторые ее усовершенствования, например расширенный размер

пакета, за счет чего повышается эффективная пропускная способность приложений, работающих поверх Bluetooth Smart.

- *Возможность использования канала Wi-Fi* (802.11b/g/n/ac) *со скоростью до 24 Мбит/с*, при этом канал Bluetooth служит для управления.

Выводы

Локальные сети на разделяемой среде представляют собой наиболее простой и дешевый в реализации тип локальных сетей. Основной недостаток разделяемых локальных сетей состоит в плохой масштабируемости, так как при увеличении числа узлов сети снижается доля пропускной способности, приходящаяся на каждый узел.

Уровень MAC отвечает за доступ к разделяемой среде и отправку через нее кадров.

Протокол LLC обеспечивает для протоколов верхних уровней нужное качество транспортных услуг, передавая кадры либо дейтаграммным способом, либо с помощью процедур с установлением соединения и восстановлением кадров.

В технологии Ethernet на разделяемой среде применяется случайный метод доступа CSMA/CD, который очень прост в реализации.

Коллизия — это ситуация, когда две станции одновременно пытаются передать кадр данных через общую среду. Наличие коллизий — неотъемлемое свойство сетей Ethernet, являющееся следствием принятого случайного метода доступа.

В зависимости от типа физической среды стандарт IEEE 802.3 определяет различные спецификации Ethernet со скоростью 10 Мбит/с: 10Base-5, 10Base-2, 10Base-T, FOIRL, 10Base-FL, 10Base-FB.

Стандарты IEEE 802.11 являются основными стандартами беспроводных локальных сетей. Существует несколько вариантов спецификаций физического уровня 802.11, отличающихся диапазоном используемых частот (2,4 и 5 ГГц), а также методом кодирования и мультиплексирования (FHSS, DSSS, OFDM).

Метод доступа 802.11 является комбинацией случайного метода доступа с предотвращением коллизий (DCF) и централизованного детерминированного метода доступа с опросом (PCF). Гибкое применение режимов DCF и PCF позволяет обеспечить поддержку показателей QoS для синхронного и асинхронного трафиков.

Персональные сети (PAN) предназначены для взаимодействия принадлежащих одному владельцу устройств на небольшом расстоянии, обычно в радиусе от 10 до 100 м. Персональные сети должны обеспечивать как фиксированный доступ, например в пределах дома, так и мобильный, когда владелец устройств перемещается вместе с ними между помещениями или городами.

Сегодня самой популярной технологией PAN является Bluetooth, в которой используется концепция пикосети, объединяющей до 255 устройств, но только восемь из них могут в каждый момент времени быть активными.

Для чувствительного к задержкам трафика сеть Bluetooth поддерживает синхронные каналы, ориентированные на соединение (SCO), а для эластичного — асинхронные каналы, не ориентированные на соединение (ACL).

Контрольные вопросы

1. Выберите утверждения, корректно описывающие особенности метода доступа технологии Ethernet:
   а) узел обязан «прослушивать» разделяемую среду;
   б) узел может передать свой кадр в разделяемую среду в любой момент времени независимо от того, занята среда или нет;
   в) узел ожидает подтверждения приема переданного кадра от узла назначения в течение некоторого времени, а в случае истечения этого времени повторяет передачу.
2. Чем объясняется, что минимальный размер поля данных кадра Ethernet выбран равным 46 байт? Варианты ответов:
   а) для предотвращения монопольного захвата среды узлом;
   б) для устойчивого распознавания коллизий;
   в) для сокращения накладных расходов.
3. К какому типу относится MAC-адрес 01:80:C2:00:00:08? Варианты ответов:
   а) групповой;
   б) индивидуальный;
   в) локальный;
   г) централизованный.
4. Как скорость передачи данных технологии Ethernet на разделяемой среде влияет на максимальный диаметр сети? Варианты ответов:
   а) чем выше скорость передачи, тем меньше максимальный диаметр сети;
   б) чем выше скорость передачи, тем больше максимальный диаметр сети;
   в) не влияет.
5. К чему приводит наличие скрытого терминала в сети IEEE 802.11? Варианты ответов:
   а) к нарушению связности сети;
   б) к повышению уровня помех в радиосреде;
   в) к более частому возникновению коллизий.

ГЛАВА 12 Коммутируемые сети Ethernet

Мост как предшественник и функциональный аналог коммутатора

Современные коммутаторы Ethernet являются наследниками мостов локальных сетей, которые широко использовались в сетях Ethernet и Token Ring на разделяемой среде. Более того, коммутаторы Ethernet по-прежнему функционально очень близки к вышедшим из употребления мостам, так как базовый алгоритм работы коммутатора и моста является одним и тем же и определяется одним стандартом IEEE 802.1D, хотя по традиции во всех новых стандартах IEEE, описывающих свойства коммутаторов, употребляется термин «коммутатор», а не «мост». Основное отличие коммутатора от моста состоит в большем количестве портов (мост, как правило, имел два порта, что и послужило поводом для его названия — мост между двумя сегментами) и более высокой производительности. Далее мы будем использовать эти термины как синонимы, выбирая нужный в зависимости от контекста.

Коммутаторы являются сегодня основным типом коммуникационных устройств, применяемых для построения локальных сетей. Коммутаторы отличаются внутренней архитектурой и конструктивным исполнением.

Коммутируемые локальные сети вытеснили локальные сети на разделяемой среде. Их успех оказал решающее влияние на эволюцию Ethernet — в новых скоростных версиях Ethernet, таких как 10G и 100G Ethernet, стандарт IEEE 802.3 описывает работу узлов только в коммутируемой среде.

Логическая структуризация сетей и мосты

Мост локальной сети (LAN bridge), или просто **мост**, появился как средство построения крупных локальных сетей на разделяемой среде. Мост объединяет две или более разделяемые среды в единую сеть, при этом передача кадров между узлами каждой из объединяемых сред происходит по стандартным правилам изолированной разделяемой среды. Мост отвечает только за передачу кадров между объединенными средами, которые называются **сегментами локальной сети**.

В сети Ethernet требование использовать единую разделяемую среду приводит к двум очень жестким ограничениям:

- общий диаметр сети не может быть больше 2500 м;
- количество узлов не может превышать 1024 (для сетей Ethernet на коаксиале это ограничение еще жестче).

На практике из-за главной проблемы разделяемой среды — дефицита пропускной способности — количество узлов в сетях Ethernet на разделяемой среде никогда не приближается к 1024.

Качественная картина зависимости задержек доступа к разделяемой среде от коэффициента использования среды показана на рис. 12.1.

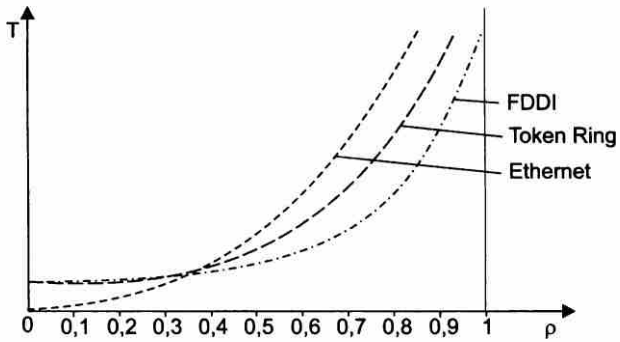


Рис. 12.1. Задержки доступа к среде передачи данных для технологий Ethernet, Token Ring и FDDI

Как видно из рисунка, всем технологиям присуща качественно одинаковая картина экспоненциального роста величины задержек доступа при увеличении коэффициента использования сети. Количество узлов, при которых коэффициент использования сети начинает приближаться к опасной границе, зависит от типа функционирующих в узлах приложений. Для сетей Ethernet со скоростью 10 Мбит/с считалось, что 30 узлов — это вполне приемлемое число для одного разделяемого сегмента, так что для построения крупной сети нужны были принципиально новые решения.

Ограничения, возникающие из-за использования единой разделяемой среды, можно преодолеть, выполнив *логическую структуризацию сети*, то есть сегментировав единую разделяемую среду на несколько и соединив полученные сегменты сети некоторым коммуникационным устройством, которое не передает данные побитно, как повторитель, а буферизует кадры и передает их затем в тот или иной сегмент (или сегменты) в зависимости от адреса назначения кадра (рис. 12.2). То есть такие сегменты работают в соответствии с обобщенным алгоритмом коммутации, рассмотренным в главе 2.

Нужно отличать логическую структуризацию от физической. Например, концентраторы стандарта 10Base-T позволяют построить сеть, состоящую из нескольких сегментов кабеля на витой паре, но это — физическая структуризация, так как логически все эти сегменты представляют собой единую разделяемую среду. Логическая структуризация сети с помощью мостов/коммутаторов является первым шагом на пути *виртуализации* сети, так как пользователям отдельного логического сегмента предоставляется виртуальный ресурс — коммуникационная среда с определенной пропускной способностью.

Логическая структуризация локальной сети позволяет решить несколько задач, основные из которых — повышение производительности, гибкости и безопасности, а также улучшение управляемости сети.

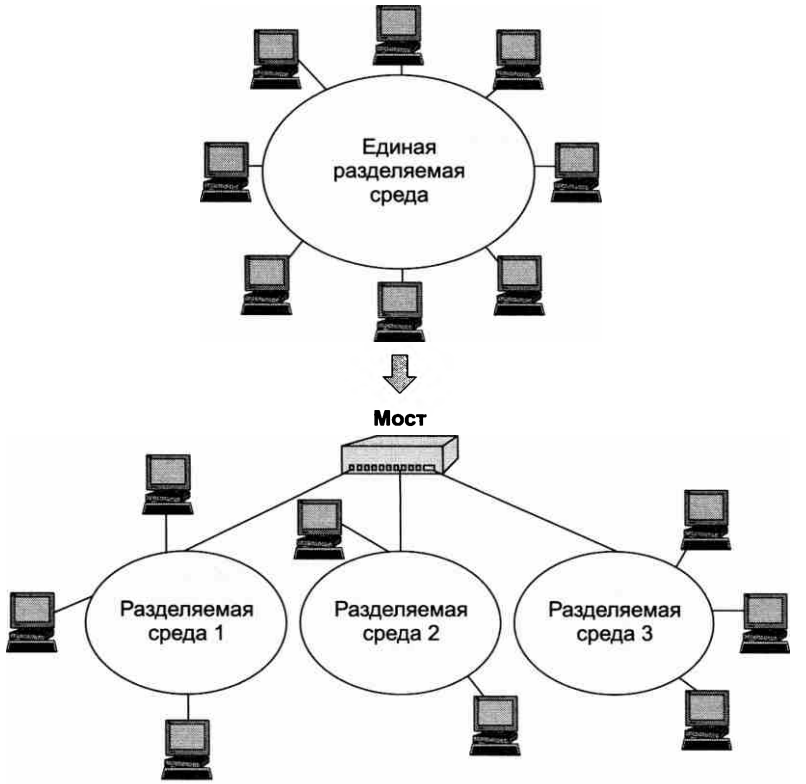


Рис. 12.2. Логическая структуризация сети

Повышение производительности сети, разделенной мостом на сегменты, происходит из-за того, что среда каждого сегмента разделяется теперь между меньшим числом конечных узлов. В примере на рис. 12.2 при разделении общей среды на три сегмента максимальное количество узлов, разделяющих среду, снизилось с 8 до 3. Как правило, разбиение на сегменты выполняется так, чтобы межсегментный трафик был небольшим, этого можно добиться, например, если каждый сегмент снабдить собственным сервером, обслуживающим запросы компьютеров сегмента.

При построении сети как совокупности сегментов каждый из них может быть адаптирован к специфическим потребностям рабочей группы или отдела. Это означает *повышение гибкости сети*. Процесс разбиения сети на логические сегменты можно рассматривать и в обратном направлении, как процесс создания большой сети из уже имеющихся небольших сетей.

Устанавливая различные логические фильтры на мостах/коммутаторах, можно контролировать доступ пользователей к ресурсам других сегментов, чего не позволяют делать повторители. Так достигается *повышение безопасности данных*.

Побочным эффектом снижения трафика и повышения безопасности данных является упрощение управления сетью, то есть *улучшение управляемости сети*. Проблемы очень часто локализуются внутри сегмента. Сегменты образуют логические домены управления сетью.

Как мосты, так и коммутаторы продвигают кадры на основании одного и того же алгоритма, а именно **алгоритма прозрачного моста**, описанного в стандарте IEEE 802.1D.

Алгоритм прозрачного моста IEEE 802.1D

Слово «прозрачный» в названии *алгоритм прозрачного моста* отражает тот факт, что конечные узлы сети функционируют, «не замечая» присутствия в сети мостов.

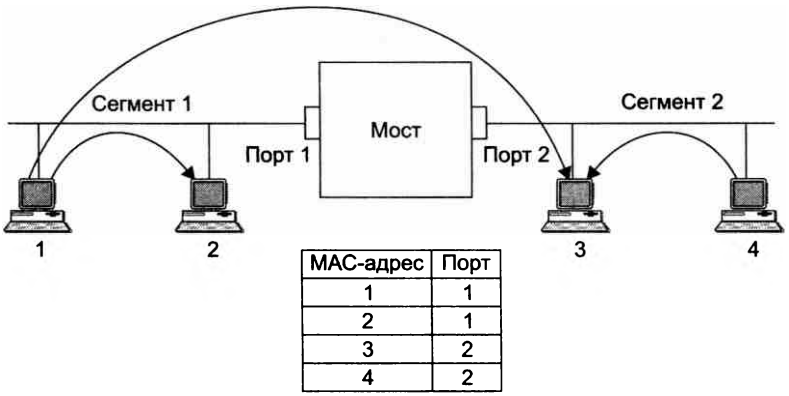
Так как алгоритм прозрачного моста остался единственным актуальным алгоритмом мостов, то в дальнейшем мы будем опускать термин «прозрачный», подразумевая именно этот тип алгоритма работы моста/коммутатора.

Мост строит свою таблицу продвижения (адресную таблицу) на основании пассивного наблюдения за трафиком, циркулирующим в подключенных к его портам сегментах. При этом мост учитывает адреса источников кадров данных, поступающих на его порты. По адресу источника кадра мост делает вывод о принадлежности узла-источника тому или иному сегменту сети.

ВНИМАНИЕ

Каждый порт моста работает как конечный узел своего сегмента, за одним исключением — порт моста может не иметь собственного MAC-адреса. Порты мостов не нуждаются в адресах для продвижения кадров, так как они работают в неразборчивом режиме захвата кадров, когда все поступающие на порт кадры независимо от их адреса назначения запоминаются на время в буферной памяти. Работая в неразборчивом режиме, мост «слушает» весь трафик, передаваемый в присоединенных к нему сегментах, и использует проходящие через него кадры для изучения топологии сети и построения таблицы продвижения. В том случае, когда порт моста/коммутатора имеет собственный MAC-адрес, он используется для целей, отличных от продвижения кадров, чаще всего — для удаленного управления портом; в этом случае порт представляет собой конечный узел сети и кадры протокола управления адресуются непосредственно ему.

Рассмотрим процесс автоматического создания таблицы продвижения моста и ее использования на примере простой сети, представленной на рис. 12.3.



MAC-адрес	Порт
1	1
2	1
3	2
4	2

Рис. 12.3. Принцип работы прозрачного моста/коммутатора

Мост соединяет два сетевых сегмента. Сегмент 1 составляют компьютеры, подключенные с помощью одного отрезка коаксиального кабеля к порту 1 моста, а сегмент 2 — компьютеры, подключенные с помощью другого отрезка коаксиального кабеля к порту 2 моста. В исходном состоянии мост не знает о том, компьютеры с какими MAC-адресами подключены к каждому из его портов. В этой ситуации мост просто передает любой захваченный и буферизованный кадр на *все* свои порты, за исключением того порта, от которого этот кадр получен. В нашем примере у моста только два порта, поэтому он передает кадры с порта 1 на порт 2, и наоборот. Отличие работы моста в этом режиме от повторителя заключается в том, что он передает кадр, предварительно буферизуя его, а не бит за битом, как это делает повторитель. Буферизация отменяет логику работы всех сегментов как единой разделяемой среды. Когда мост собирается передать кадр с сегмента на сегмент, например с сегмента 1 на сегмент 2, он, как обычный конечный узел, пытается получить доступ к разделяемой среде сегмента 2 по правилам алгоритма доступа, в данном примере — по правилам алгоритма CSMA/CD.

Одновременно с передачей кадра на все порты мост изучает адрес источника кадра и делает запись о его принадлежности к тому или иному сегменту в своей **адресной таблице**. Эту таблицу также называют **таблицей фильтрации**, или **продвижения**. Например, получив на порт 1 кадр от компьютера 1, мост делает первую запись в своей адресной таблице:

MAC-адрес 1 — порт 1.

Эта запись означает, что компьютер, имеющий MAC-адрес 1, принадлежит сегменту, подключенному к порту 1 коммутатора. Если все четыре компьютера данной сети проявляют активность и посылают друг другу кадры, то скоро мост построит полную адресную таблицу сети, состоящую из четырех записей — по одной записи на узел (см. рис. 12.3).

При каждом поступлении кадра на порт моста он прежде всего пытается найти адрес назначения кадра в адресной таблице. Продолжим рассмотрение действий моста на примере (см. рис. 12.3).

1. При получении кадра, направленного от компьютера 1 компьютеру 3, мост просматривает адресную таблицу на предмет совпадения адреса в какой-либо из ее записей с адресом назначения — MAC-адресом 3. Запись с искомым адресом имеется в адресной таблице.
2. Мост выполняет второй этап анализа таблицы — проверяет, находятся ли компьютеры с адресами источника и назначения в одном сегменте. В примере компьютер 1 (MAC-адрес 1) и компьютер 3 (MAC-адрес 3) находятся в разных сегментах. Следовательно, мост выполняет операцию **продвижения** (forwarding) кадра — передает кадр на порт 2, ведущий в сегмент получателя, получает доступ к сегменту и передает туда кадр.
3. Если бы оказалось, что компьютеры принадлежали одному сегменту, то кадр просто был бы удален из буфера. Такая операция называется **фильтрацией** (filtering).
4. Если бы запись о MAC-адресе 3 отсутствовала в адресной таблице, то есть, другими словами, *адрес назначения был неизвестен* мосту, то он передал бы кадр на все свои порты, кроме порта — источника кадра, как и на начальной стадии процесса обучения.

Процесс обучения моста никогда не заканчивается и происходит одновременно с продвижением и фильтрацией кадров. Мост постоянно следит за адресами источника буферизуемых кадров, чтобы автоматически приспосабливаться к изменениям, происходящим в сети, — перемещениям компьютеров из одного сегмента сети в другой, отключению и появлению новых компьютеров.

Входы адресной таблицы могут быть динамическими, создаваемыми в процессе *самообучения* моста, и статическими, создаваемыми *вручную* администратором сети. **Статические записи** не имеют срока жизни, что дает администратору возможность влиять на работу моста, например ограничивая передачу кадров с определенными адресами из одного сегмента в другой.

Динамические записи имеют срок жизни — при создании или обновлении записи в адресной таблице с ней связывается отметка времени. По истечении определенного тайм-аута запись помечается как недействительная, если за это время мост не принял ни одного кадра с данным адресом в поле адреса источника. Это дает возможность мосту автоматически реагировать на перемещения компьютера из сегмента в сегмент — при его отключении от старого сегмента запись о принадлежности компьютера к этому сегменту со временем вычеркивается из адресной таблицы. После подключения компьютера к другому сегменту его кадры начнут попадать в буфер моста через другой порт и в адресной таблице появится новая запись, соответствующая текущему состоянию сети.

Кадры с широковещательными и групповыми MAC-адресами, как и кадры с неизвестными адресами назначения, передаются мостом на все его порты. Такой режим распространения кадров называется **затоплением сети** (flooding). Наличие мостов в сети не препятствует распространению широковещательных и групповых кадров по всем сегментам сети. Однако это является достоинством только тогда, когда такой адрес выработан корректно работающим узлом.

Нередко в результате каких-либо программных или аппаратных сбоев протокол верхнего уровня или сетевой адаптер начинает работать некорректно, а именно постоянно с высокой интенсивностью генерировать кадры с широковещательным адресом. Мост в соответствии со своим алгоритмом передает ошибочный трафик во все сегменты. Такая ситуация называется **широковещательным штормом** (broadcast storm).

К сожалению, мосты не защищают сети от широковещательного шторма, во всяком случае, по умолчанию, как это делают маршрутизаторы (вы познакомитесь с этим свойством маршрутизаторов в части IV). Максимум, что может сделать администратор с помощью коммутатора для борьбы с широковещательным штормом, — установить для каждого порта моста предельно допустимую интенсивность передачи кадров с широковещательным адресом. Но при этом нужно точно знать, какая интенсивность является нормальной, а какая — ошибочной. При смене протоколов ситуация в сети может измениться, и то, что вчера считалось ошибочным, сегодня может оказаться нормой.

На рис. 12.4 показана типичная структура моста. Функции доступа к среде при приеме и передаче кадров выполняют микросхемы MAC, которые идентичны микросхемам сетевого адаптера.

Протокол, реализующий алгоритм коммутатора, располагается между уровнями MAC и LLC.

На рис. 12.5 показана копия экрана терминала с адресной таблицей моста.

Из выводимой на экран адресной таблицы видно, что сеть состоит из двух сегментов — LAN *A* и LAN *B*. В сегменте LAN *A* имеются по крайней мере три станции, а в сегменте LAN *B* — две станции. Четыре адреса, помеченные звездочками, являются статическими, то есть назначенными администратором вручную. Адрес, помеченный плюсом, является динамическим адресом с истекшим сроком жизни.

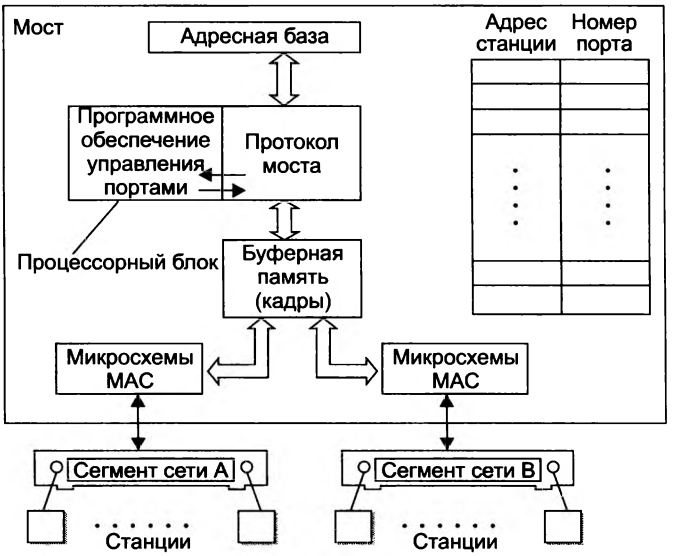


Рис. 12.4. Структура моста/коммутатора

```
                          Forwarding Table                    Page 1 of 1

Address          Dispn    Address          Dispn    Address          Dispn
 00608CB17E58    LAN B     0000810298D6    LAN A     02070188ACA     LAN A
 00008101C4DF    LAN B   + 000081016A52    LAN A   * 010081000100    Flood
* 010081000101   Discard  * 0180C2000000   Discard   * 000081FFD166    Flood

                                  Статус адреса:
                                  срок жизни записи истек

Exit      Next Page      Prev Page      Edit Table      Search Item      Go Page
          + Unlearned    * Static    Total Entries = 9    Static Entries = 4
          Use cursor keys to choose option. Press <RETURN> to select.
                    Press <CTRL> <P> to return to Main Menu
```

Рис. 12.5. Адресная таблица коммутатора

Таблица имеет поле *Dispn* — «disposition» (это «распоряжение» мосту о том, какую операцию нужно проделать с кадром, имеющим данный адрес назначения). Обычно при автоматическом составлении таблицы в этом поле ставится условное обозначение порта назначения, при ручном же задании адреса в это поле можно внести нестандартную операцию обработки кадра. Например, операция *Flood* (затопление) заставляет мост распространять кадр в широковещательном режиме, несмотря на то что его адрес назначения не является широковещательным. Операция *Discard* (отбросить) говорит мосту, что кадр с таким адресом не нужно передавать на порт назначения. Вообще говоря, операции, задаваемые в поле *Dispn*, определяют особые условия фильтрации кадров, дополняющие стандартные условия их распространения. Такие условия обычно называют **пользовательскими фильтрами**, мы их рассмотрим немного позже в разделе «Фильтрация трафика» главы 13.

Топологические ограничения при применении мостов в локальных сетях

Серьезным ограничением функциональных возможностей мостов и коммутаторов является отсутствие поддержки петлеобразных конфигураций сети.

Рассмотрим это ограничение на примере сети, показанной на рис. 12.6.

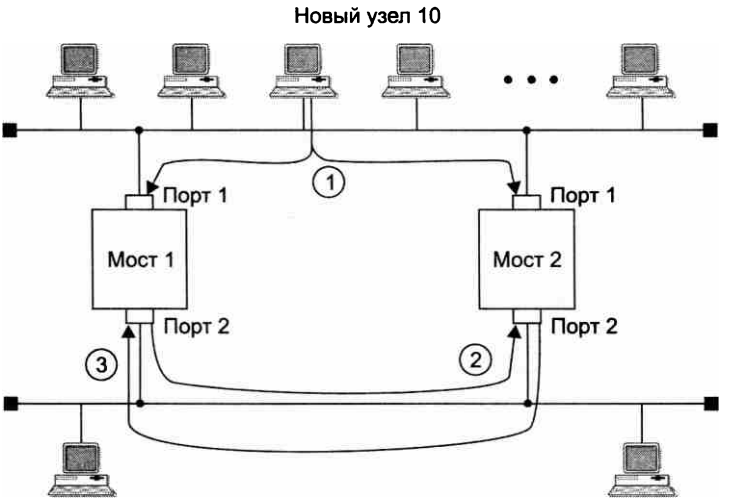


Рис. 12.6. Влияние замкнутых маршрутов на работу коммутаторов

Два сегмента Ethernet параллельно соединены двумя мостами так, что образовалась петля. Пусть новая станция с MAC-адресом 123 впервые начинает работу в данной сети. Обычно начало работы любой операционной системы сопровождается рассылкой широковещательных кадров, в которых станция заявляет о своем существовании и одновременно ищет серверы сети.

На этапе 1 станция посылает первый кадр с широковещательным адресом назначения и адресом источника 123 в свой сегмент. Кадр попадает как в мост 1, так и в мост 2. В обоих мостах новый адрес источника 123 заносится в адресную таблицу с пометкой о его принадлежности сегменту 1, то есть создается новая запись вида:

MAC-адрес 123 — Порт 1.

Так как адрес назначения широковещательный, то каждый мост должен передать кадр на сегмент 2. Эта передача происходит поочередно в соответствии с методом случайного доступа технологии Ethernet. Пусть первым доступ к сегменту 2 получает мост 1 (этап 2 на рис. 12.6). При появлении кадра на сегменте 2 мост 2 принимает его в свой буфер и обрабатывает. Он видит, что адрес 123 уже есть в его адресной таблице, но пришедший кадр является более свежим, и он решает, что адрес 123 принадлежит сегменту 2, а не 1. Поэтому мост 2 корректирует содержимое базы и делает запись о том, что адрес 123 принадлежит сегменту 2:

MAC-адрес 123 — Порт 2.

Аналогично поступает мост 1, когда мост 2 передает свою копию кадра на сегмент 2.

Далее перечислены последствия наличия петли в сети.

- «Размножение» кадра, то есть появление нескольких его копий (в данном случае — двух, но если бы сегменты были соединены тремя мостами — то трех и т. д.).
- Бесконечная циркуляция обеих копий кадра по петле в противоположных направлениях, а значит, засорение сети ненужным трафиком.
- Постоянная перестройка мостами своих адресных таблиц, так как кадр с адресом источника 123 будет появляться то на одном порту, то на другом.

В целях исключения всех этих нежелательных эффектов мосты/коммутаторы нужно применять так, чтобы между логическими сегментами не было петель, то есть строить с помощью коммутаторов только древовидные структуры, гарантирующие наличие единственного пути между любыми двумя сегментами. Тогда кадры от каждой станции будут поступать на мост/коммутатор всегда с одного и того же порта и коммутатор сможет правильно решать задачу выбора рационального маршрута в сети.

В небольших сетях сравнительно легко гарантировать наличие одного и только одного пути между двумя сегментами. Но когда количество соединений возрастает, то вероятность непреднамеренного образования петли оказывается высокой.

Возможна и другая причина возникновения петель. Так, для повышения надежности желательно иметь между мостами/коммутаторами резервные связи, которые не участвуют в нормальной работе основных связей по передаче информационных кадров станций, но при отказе какой-либо основной связи образуют новую связную рабочую конфигурацию без петель.

Избыточные связи необходимо блокировать, то есть переводить их в неактивное состояние. В сетях с простой топологией эта задача решается вручную путем блокирования соответствующих портов мостов/коммутаторов. В больших сетях со сложными связями используются алгоритмы, которые позволяют решать задачу обнаружения петель автоматически. Наиболее известным из них является стандартный **алгоритм покрывающего дерева** (Spanning Tree Algorithm, STA), который детально рассмотрен в главе 13.

Коммутаторы

Параллельная коммутация

При появлении в конце 80-х — начале 90-х годов быстрых протоколов, производительных персональных компьютеров, мультимедийной информации и разделении сети на большое количество сегментов классические *мосты* перестали справляться с работой. Обслуживание потоков кадров между теперь уже несколькими портами с помощью одного процессорного блока требовало значительного повышения быстродействия процессора, а это довольно дорогостоящее решение.

Более эффективным оказалось решение, которое и «породило» коммутаторы: для обслуживания потока, поступающего на каждый порт, в устройство ставился отдельный специализированный процессор, который реализовывал алгоритм прозрачного моста. По сути,

коммутатор — это мультипроцессорный мост, способный параллельно продвигать кадры сразу между всеми парами своих портов. Но если при добавлении процессорных блоков компьютер не перестали называть компьютером, а добавили только прилагательное «мультипроцессорный», то с мультипроцессорными мостами произошла метаморфоза — во многом по маркетинговым причинам они превратились в коммутаторы. Нужно отметить, что помимо процессоров портов коммутатор имеет центральный процессор, который координирует работу портов, отвечая за построение общей таблицы продвижения, а также поддерживая функции конфигурирования и управления коммутатором.

Со временем коммутаторы вытеснили из локальных сетей классические однопроцессорные мосты. Основная причина этого — существенно более высокая производительность, с которой коммутаторы передают кадры между сегментами сети. Если мосты могли даже замедлять работу сети, то коммутаторы всегда выпускаются с процессорами портов, способными передавать кадры с той максимальной скоростью, на которую рассчитан протокол. Ну а добавление к этому возможности параллельной передачи кадров между портами предопределило судьбу и мостов, и коммутаторов.

Производительность коммутаторов на несколько порядков выше, чем мостов, — коммутаторы могут передавать до нескольких десятков, а иногда и сотен миллионов кадров в секунду, в то время как мосты обычно обрабатывали 3–5 тысяч кадров в секунду.

За время своего существования уже без конкурентов-мостов коммутаторы вобрали в себя многие дополнительные функции, родившиеся в результате естественного развития сетевых технологий. К этим функциям относятся, например, поддержка виртуальных сетей (VLAN), агрегирование линий связи, приоритезация трафика и т. п. Развитие технологии производства заказных микросхем также способствовало успеху коммутаторов, в результате процессоры портов сегодня обладают такой вычислительной мощностью, которая позволяет им быстро реализовывать весьма сложные алгоритмы обработки трафика, например выполнять его классификацию и профилирование.

Основной причиной повышения производительности сети при использовании коммутатора является *параллельная* обработка нескольких кадров.

Этот эффект иллюстрирует рис. 12.7, на котором показана идеальная в отношении производительности ситуация, когда четыре порта из восьми передают данные с максимальной для протокола Ethernet скоростью в 10 Мбит/с. Причем они передают эти данные на остальные четыре порта коммутатора не конфликтуя: потоки данных между узлами сети распределились так, что для каждого принимающего кадры порта есть свой выходной порт. Если коммутатор успевает обрабатывать входной трафик при максимальной интенсивности поступления кадров на входные порты, то общая производительность коммутатора в приведенном примере составит $4 \times 10 = 40$ Мбит/с, а при обобщении примера для N портов — $(N/2) \times 10$ Мбит/с. В таком случае говорят, что *коммутатор предоставляет каждой станции или сегменту, подключенному к его портам, выделенную пропускную способность протокола*.

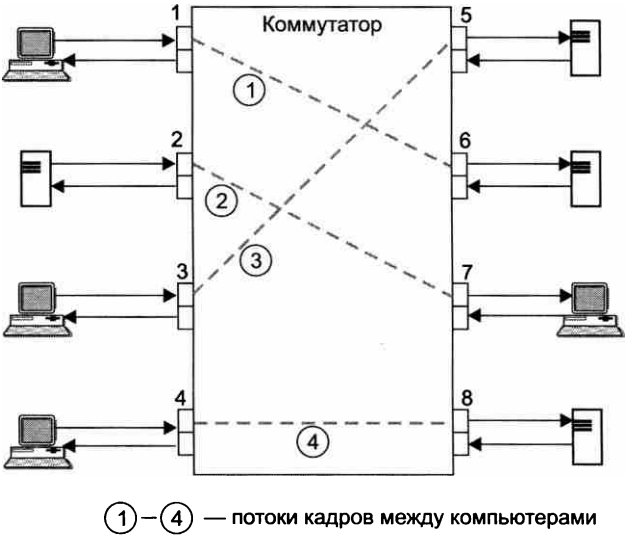


Рис. 12.7. Параллельная передача кадров коммутатором

Естественно, что в сети не всегда складывается описанная ситуация. Если двум станциям, например станциям, подключенным к портам 3 и 4, одновременно нужно записывать данные на один и тот же сервер, подключенный к порту 8, то коммутатор не сможет выделить каждой станции по 10 Мбит/с, так как порт 8 не в состоянии передавать данные со скоростью 20 Мбит/с. Кадры станций будут ожидать во внутренних очередях входных портов 3 и 4, когда освободится порт 8 для передачи очередного кадра. Очевидно, хорошим решением для такого распределения потоков данных было бы подключение сервера к более высокоскоростному порту, например Fast Ethernet или Gigabit Ethernet.

Дуплексный режим работы

Технология коммутации сама по себе не имеет непосредственного отношения к методу доступа к среде, который используется портами коммутатора. При подключении к порту коммутатора сегмента, представляющего собой разделяемую среду, данный порт, как и все остальные узлы такого сегмента, должен поддерживать полудуплексный режим.

Однако когда к каждому порту коммутатора подключен не сегмент, а только *один* компьютер, причем по двум физически раздельным каналам, как это происходит почти во всех стандартах Ethernet, кроме коаксиальных версий Ethernet, ситуация становится не такой однозначной. Порт может работать как в обычном полудуплексном режиме, так и в дуплексном.

В **полудуплексном режиме** работы порт коммутатора по-прежнему распознает коллизии. Доменом коллизий в этом случае является участок сети, включающий передатчик коммутатора, приемник коммутатора, передатчик сетевого адаптера компьютера, приемник сетевого адаптера компьютера и две витые пары, соединяющие передатчики с приемниками.

Коллизия возникает, когда передатчики порта коммутатора и сетевого адаптера одновременно или почти одновременно начинают передачу своих кадров.

В **дуплексном режиме** одновременная передача данных передатчиком порта коммутатора и сетевого адаптера коллизией не считается. В принципе, это достаточно естественный режим работы для отдельных дуплексных каналов передачи данных, и он всегда использовался в протоколах глобальных сетей. При дуплексной связи порты Ethernet стандарта 10 Мбит/с могут передавать данные со скоростью 20 Мбит/с — по 10 Мбит/с в каждом направлении.

Долгое время коммутаторы Ethernet сосуществовали в локальных сетях с концентраторами Ethernet: на концентраторах строились нижние уровни сети здания, такие как сети рабочих групп и отделов, а коммутаторы служили для объединения этих сегментов в общую сеть.

Постепенно коммутаторы стали применяться и на нижних этажах, вытесняя концентраторы, так как цены коммутаторов постоянно снижались, а их производительность росла (за счет поддержки более скоростных версий технологии Ethernet, то есть Fast Ethernet со скоростью 100 Мбит/с, Gigabit Ethernet со скоростью 1 Гбит/с, 10G Ethernet со скоростью 10 Гбит/с и 100G Ethernet со скоростью 100 Гбит/с — эти версии рассматриваются далее в разделе «Скоростные версии Ethernet»). Этот процесс завершился вытеснением концентраторов Ethernet и переходом к полностью коммутируемым сетям, пример такой сети показан на рис. 12.8.

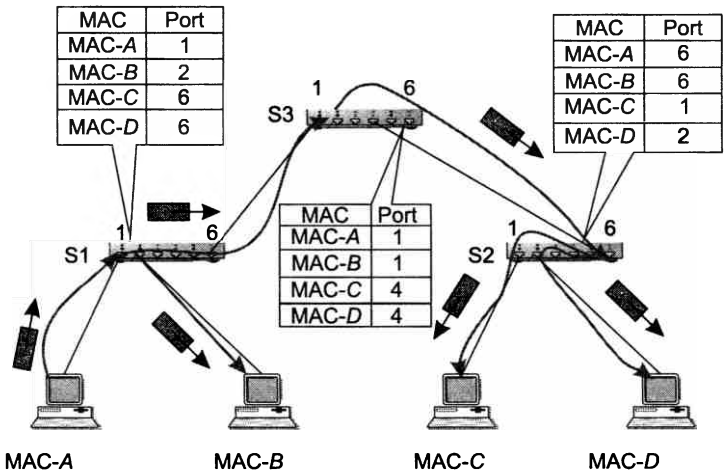


Рис. 12.8. Полностью коммутируемая сеть Ethernet

В полностью коммутируемой сети Ethernet все порты работают в дуплексном режиме, а продвижение кадров осуществляется на основе MAC-адресов.

При разработке технологий Fast Ethernet и Gigabit Ethernet дуплексный режим стал одним из двух полноправных стандартных режимов работы узлов сети. Однако практика применения первых коммутаторов с портами Gigabit Ethernet показала, что они практически всегда применяются в дуплексном режиме для взаимодействия с другими коммутаторами или высокоскоростными сетевыми адаптерами. Поэтому при разработке версий стандартов 10G и 100G Ethernet его разработчики не стали создавать версию для работы в полудуплексном режиме, окончательно закрепив уход разделяемой среды из технологии Ethernet.

Неблокирующие коммутаторы

Как уже отмечалось, высокая производительность является одним из главных достоинств коммутаторов. С понятием производительности тесно связано понятие неблокирующего коммутатора.

Коммутатор называют **неблокирующим**, если он может передавать кадры через свои порты с той же скоростью, с которой они на них поступают.

Когда говорят, что коммутатор может поддерживать *устойчивый неблокирующий режим работы*, то имеют в виду, что коммутатор передает кадры со скоростью их поступления в течение произвольного промежутка времени. Для поддержания подобного режима нужно таким образом распределить потоки кадров по выходным портам, чтобы, во-первых, порты справлялись с нагрузкой, во-вторых, коммутатор мог всегда в среднем передать на выходы столько кадров, сколько их поступило на входы. Если же входной поток кадров (просуммированный по всем портам) в среднем будет превышать выходной поток кадров (также просуммированный по всем портам), то кадры будут накапливаться в буферной памяти коммутатора и при переполнении — просто отбрасываться.

Для поддержания устойчивого неблокирующего режима работы коммутатора необходимо, чтобы его производительность удовлетворяла условию $C_k = (\Sigma C_{pi})/2$, где C_k — производительность коммутатора, C_{pi} — максимальная производительность протокола, поддерживаемого i-м портом коммутатора.

В этом соотношении под производительностью коммутатора в целом понимается его способность продвигать на передатчики всех своих портов определенное количество кадров, принимаемых от приемников всех своих портов.

В суммарной производительности портов каждый проходящий кадр учитывается дважды, как входящий и как выходящий, а так как в устойчивом режиме входной трафик равен выходному, то минимально достаточная производительность коммутатора для поддержки неблокирующего режима равна половине суммарной производительности портов. Если порт, например, стандарта Ethernet со скоростью 10 Мбит/с работает в полудуплексном режиме, то производительность порта C_{pi} равна 10 Мбит/с, а если в дуплексном — 20 Мбит/с.

Иногда говорят, что коммутатор поддерживает *мгновенный неблокирующий режим*. Это означает, что он может принимать и обрабатывать кадры от всех своих портов на максимальной скорости протокола независимо от того, обеспечиваются ли условия устойчивого равновесия между входным и выходным трафиком. Правда, обработка некоторых кадров при этом может быть неполной — при занятости выходного порта кадр помещается в буфер коммутатора.

Для поддержки мгновенного неблокирующего режима коммутатор должен обладать большей собственной производительностью, а именно она должна быть равна суммарной производительности его портов: $C_k = \Sigma C_{pi}$.

Приведенные соотношения справедливы для портов с любыми скоростями, то есть портов стандартов Ethernet со скоростью 10 Мбит/с, Fast Ethernet, Gigabit Ethernet, 10G и 100G Ethernet.

Способы, которыми осуществляется способность коммутатора поддерживать неблокирующий режим, могут быть разными. Необходимым требованием является умение процессора порта обрабатывать потоки кадров с максимальной для физического уровня этого порта скоростью. В главе 11 мы подсчитали, что максимальная производительность порта Ethernet стандарта 10 Мбит/с равна 14 880 кадров (минимальной длины) в секунду. Это означает, что процессоры портов Ethernet стандарта 10 Мбит/с неблокирующего коммутатора должны поддерживать продвижение кадров со скоростью 14 880 кадров в секунду. Как вы увидите дальше, более скоростные версии Ethernet сохраняют формат кадра Ethernet и сокращают межкадровый интервал пропорционально увеличению битовой скорости версии (то есть в 10 раз при повышении скорости в 10 раз). Поэтому максимальные значения скорости продвижения кадров также растут пропорционально росту битовой скорости, например Fast Ethernet обеспечивает максимальную скорость продвижения в 148 800 кадров в секунду, а Gigabit Ethernet — в 1 488 000 кадров в секунду. Соответственно должна расти скорость продвижения коммутатора Ethernet с высокоскоростными портами.

Однако только адекватной производительности процессоров портов недостаточно для того, чтобы коммутатор был неблокирующим. Необходимо, чтобы достаточной производительностью обладали все элементы архитектуры коммутатора, включая центральный процессор, общую память, шины, соединяющие отдельные модули между собой, саму архитектура коммутатора (наиболее распространенные архитектуры коммутаторов мы рассмотрим позже). В принципе, задача создания неблокирующего коммутатора аналогична задаче создания высокопроизводительного компьютера — в обоих случаях она решается комплексно: за счет соответствующей архитектуры объединения модулей в едином устройстве и адекватной производительности каждого отдельного модуля устройства.

Борьба с перегрузками

Даже в том случае, когда коммутатор является неблокирующим, нет гарантии того, что он во всех случаях справится с потоком кадров, направляемых на его порты. Неблокирующие коммутаторы тоже могут испытывать перегрузки и терять кадры из-за переполнения внутренних буферов.

Причина перегрузок обычно кроется не в том, что коммутатору не хватает производительности для обслуживания потоков кадров, а в ограниченной пропускной способности конкретного выходного порта, которая определяется параметрами протокола. Другими словами, какой бы производительностью коммутатор ни обладал, всегда найдется такое распределение потоков кадров, которое приведет к перегрузке коммутатора из-за ограниченной производительности выходного порта коммутатора.

Возникновение таких перегрузок является платой за отказ от применения алгоритма доступа к разделяемой среде, так как в дуплексном режиме работы портов теряется контроль над потоками кадров, направляемых конечными узлами в сеть. В полудуплексном режиме, свойственном технологиям с разделяемой средой, поток кадров регулируется самим мето-

дом доступа к разделяемой среде. При переходе на дуплексный режим узлу разрешается отправлять кадры в коммутатор всегда, когда это ему нужно, поэтому в данном режиме коммутаторы сети могут сталкиваться с перегрузками, не имея при этом никаких средств «притормаживания» потока кадров.

Таким образом, если входной трафик неравномерно распределяется между выходными портами, легко представить ситуацию, когда на какой-либо выходной порт коммутатора будет направляться трафик с суммарной средней интенсивностью большей, чем протокольный максимум. На рис. 12.9 показана как раз такая ситуация, когда на порт 3 коммутатора Ethernet от портов 1, 2, 4 и 6 направляется поток кадров размером в 64 байт с суммарной интенсивностью в 22 100 кадров в секунду. Вспомним, что максимальная скорость в кадрах в секунду для сегмента Ethernet составляет 14 880. Естественно, что когда кадры поступают в буфер порта со скоростью 22 100 кадров в секунду, а уходят со скоростью 14 880 кадров в секунду, то внутренний буфер выходного порта начинает неуклонно заполняться необработанными кадрами.

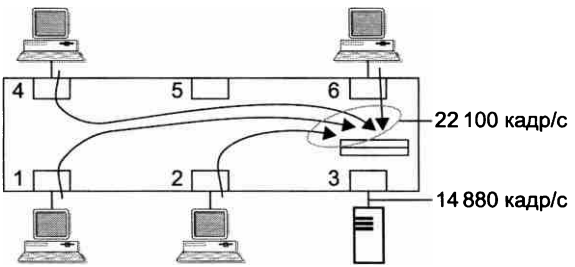


Рис. 12.9. Переполнение буфера порта из-за несбалансированности трафика

В приведенном примере нетрудно подсчитать, что при размере буфера в 100 Кбайт полное заполнение буфера произойдет через 0,22 секунды после начала работы в таком интенсивном режиме. Увеличение размера буфера до 1 Мбайт даст увеличение времени заполнения буфера до 2,2 секунд, что также неприемлемо. Проблему можно решить с помощью *средств контроля перегрузки*, которые были рассмотрены в главе 6.

Как мы знаем, существуют различные средства контроля перегрузки: управление очередями в коммутаторах, обратная связь, резервирование пропускной способности. На основе этих средств можно создать эффективную систему поддержки показателей QoS для трафика разных классов.

В этом разделе мы рассмотрим **механизм обратной связи**, который был стандартизован для сетей Ethernet в марте 1997 года как спецификация IEEE 802.3x. Механизм обратной связи 802.3x используется только в дуплексном режиме работы портов коммутатора. Этот механизм очень важен для коммутаторов локальных сетей, так как он позволяет сократить потери кадров из-за переполнения буферов.

Спецификация 802.3x вводит новый подуровень в стеке протоколов Ethernet — **подуровень управления уровня MAC**. Он располагается над уровнем MAC и является необязательным.

Кадры этого подуровня могут использоваться в различных целях, но пока в стандартах Ethernet для них определена только одна задача — приостановка передачи кадров другими узлами на определенное время.

Коммутатор использует кадр подуровня управления в том случае, когда ему нужно на время приостановить поступление кадров от соседнего узла, чтобы разгрузить свои внутренние очереди.

В качестве адреса назначения можно указывать зарезервированное для этой цели значение группового адреса 01-80-C2-00-00-01. Это удобно, когда соседний узел также является коммутатором (так как порты коммутатора не имеют уникальных MAC-адресов). Если сосед — конечный узел, можно также использовать уникальный MAC-адрес.

В поле кода операции подуровня управления указывается шестнадцатеричный код 00-01, поскольку, как уже было отмечено, пока определена только одна операция подуровня управления — она называется *PAUSE* (пауза) и имеет шестнадцатеричный код 00-01.

В поле параметров подуровня управления указывается время, на которое узел, получивший такой код, должен прекратить передачу кадров узлу, отправившему кадр с операцией *PAUSE*. Время измеряется в 512 битовых интервалах конкретной реализации Ethernet, диапазон возможных вариантов приостановки равен 0–65 535.

Как видно из описания, этот механизм обратной связи относится к типу 2 в соответствии с классификацией, приведенной в главе 6. Специфика его состоит в том, что в нем предусмотрена только одна операция — приостановка на определенное время. Обычно же в механизмах этого типа используются две операции — приостановка и возобновление передачи кадров.

Проблема, иллюстрируемая рис. 12.9, может быть решена и другим способом: применением так называемого **магистрального**, или **восходящего** (uplink), **порта**. Магистральные порты в коммутаторах Ethernet — это порты следующего уровня иерархии скорости по сравнению с портами, предназначенными для подключения пользователей. Например, если коммутатор имеет 12 портов Ethernet стандарта 10 Мбит/с, то магистральный порт должен быть портом Fast Ethernet, чтобы его скорость была достаточна для передачи до 10 потоков от входных портов. Обычно низкоскоростные порты коммутатора служат для соединения с пользовательскими компьютерами, а магистральные порты — для подключения либо сервера, к которому обращаются пользователи, либо коммутатора более высокого уровня иерархии.

На рис. 12.10 показан пример коммутатора, имеющего 24 порта стандарта Fast Ethernet со скоростью 100 Мбит/с, к которым подключены пользовательские компьютеры, и один порт стандарта Gigabit Ethernet со скоростью 1000 Мбит/с, к которому подключен сервер. При такой конфигурации коммутатора вероятность перегрузки портов существенно снижается по сравнению с вариантом, когда все порты поддерживают одинаковую скорость. Хотя возможность перегрузки по-прежнему существует, для этого необходимо, чтобы более 10 пользователей одновременно обменивались с сервером данными со средней скоростью, близкой к максимальной скорости их соединений, а такое событие достаточно маловероятно.

Из приведенного примера видно, что вероятность перегрузки портов коммутаторов зависит от распределения трафика между его портами, кроме того, понятно, что даже при хорошем соответствии скорости портов наиболее вероятному распределению трафика полностью исключить перегрузки невозможно.

Поэтому в общем случае для уменьшения потерь кадров из-за перегрузок нужно применять оба средства: подбор скорости портов для наиболее вероятного распределения трафика в сети и протокол 802.3x для снижения скорости источника трафика в тех случаях, когда перегрузки все-таки возникают.

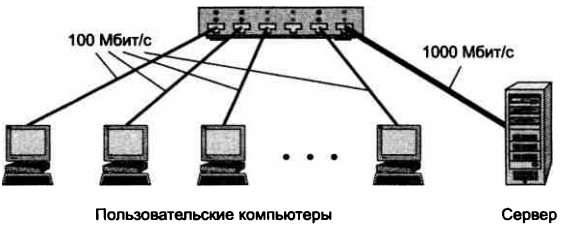


Рис. 12.10. Коммутатор рабочей группы

Скоростные версии Ethernet

Скорость 10 Мбит/с первой стандартной версии Ethernet долгое время удовлетворяла потребности пользователей локальных сетей. Однако в начале 90-х годов начала ощущаться недостаточная пропускная способность Ethernet, так как скорость обмена с сетью стала существенно меньше скорости внутренней шины компьютера. Кроме того, начали появляться новые мультимедийные приложения, гораздо более требовательные к скорости сети, чем их текстовые предшественники. В поисках решения проблемы ведущие производители сетевого оборудования начали интенсивные работы по повышению скорости Ethernet при сохранении главного достоинства этой технологии — простоты и невысокой стоимости оборудования.

Результатом стало появление новых скоростных стандартов Ethernet: Fast Ethernet (скорость 100 Мбит/с), Gigabit Ethernet (1000 Мбит/с, или 1 Гбит/с), 10G Ethernet (10 Гбит/с) и 100G Ethernet (100 Гбит/с).

Разработчикам новых скоростных стандартов Ethernet удалось сохранить основные черты классической технологии Ethernet, и прежде всего простой способ обмена кадрами без встраивания в технологию сложных контрольных процедур. Этот фактор оказался решающим в соревновании технологий локальных сетей, так как выбор пользователей всегда склонялся в пользу простого наращивания скорости сети, а не в пользу решений, связанных с более эффективным расходованием той же самой пропускной способности с помощью более сложной и дорогой технологии.

Значительный вклад в «победу» Ethernet внесли также коммутаторы локальных сетей, так как их успех привел к отказу от разделяемой среды, где технология Ethernet всегда была уязвимой из-за случайного характера метода доступа. Начиная с версии 10G Ethernet, разработчики перестали включать вариант работы на разделяемой среде в описание стандарта.

Повышение скорости работы Ethernet было достигнуто за счет улучшения качества кабелей, применяемых в компьютерных сетях, совершенствования методов кодирования данных при их передаче по витым парам (а также использования параллельных потоков данных, то есть за счет совершенствования физического уровня технологии).

Fast Ethernet

Физические уровни технологии Fast Ethernet

Разработчикам технологии Fast Ethernet удалось обеспечить ее преемственность с классической технологией Ethernet 10 Мбит/с.

Поэтому все отличия технологий Fast Ethernet и Ethernet проявляются на физическом уровне (рис. 12.11). Уровни MAC и LLC в Fast Ethernet остались абсолютно теми же, и их описывают прежние главы стандартов 802.3 и 802.2.

Рассматривая технологию Fast Ethernet, мы будем изучать только варианты ее физического уровня.

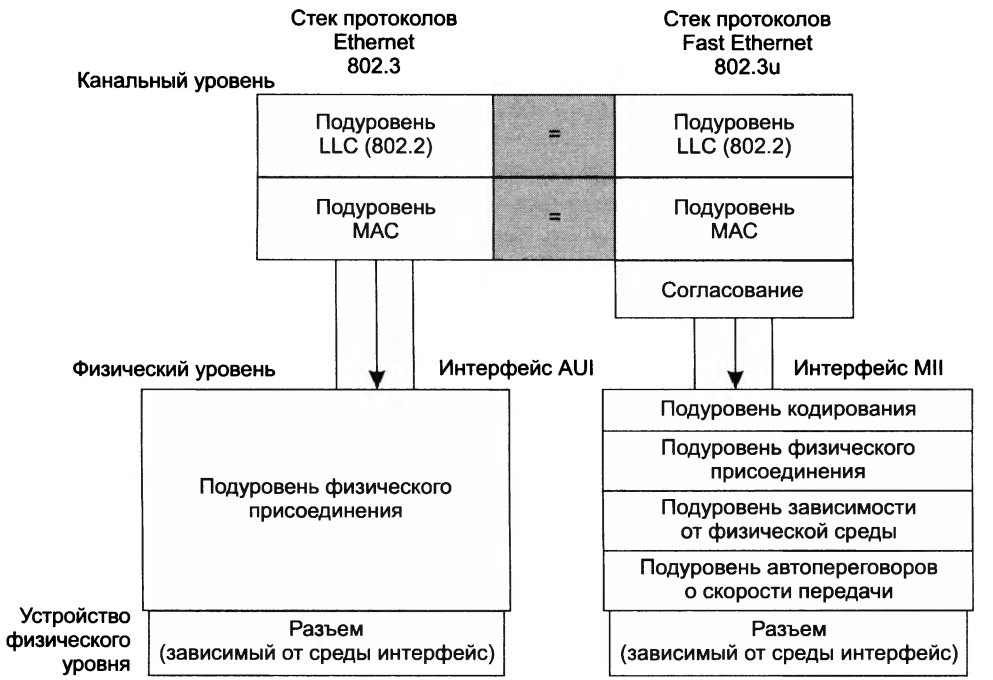


Рис. 12.11. Отличия технологий Fast Ethernet и Ethernet

Организация физического уровня технологии Fast Ethernet является модульной. Это объясняется тем, что технология Fast Ethernet изначально была рассчитана на применение различных типов физической среды и кодирования, модульность физического уровня позволяет достичь этой цели достаточно легко. Различные же варианты физической среды Ethernet 10 Мбит/с разрабатывались разными организациями в разное время, отсюда и отсутствие гибкости в построении физического уровня. Нужно подчеркнуть, что этот модульный подход был впоследствии применен и во всех других более скоростных вариантах Ethernet, включая 100G Ethernet.

Физический уровень Fast Ethernet состоит из трех модулей:

- **Независимый от среды интерфейс** (Media Independent Interface, MII). Этот интерфейс поддерживает независимый от физической среды способ обмена данными между подуровнями MAC и PHY.
- **Модуль согласования** (Reconciliation) нужен для того, чтобы уровень MAC, рассчитанный ранее на интерфейс AUI, мог работать с физическим уровнем через интерфейс MII.
- **Устройство физического уровня** (Physical Layer Device, PHY) состоит, в свою очередь, из нескольких подуровней (см. рис. 12.11):
  - подуровня кодирования данных (Physical Coding Sublayer, PCS), преобразующего поступающие от уровня MAC байты в символы логического кода, например 4B/5B;
  - подуровней физического присоединения (Physical Media Attachment, PMA) и зависимости от физической среды (Physical Media Dependent, PMD), которые обеспечивают формирование сигналов в соответствии с методом физического кодирования, например NRZI;
  - подуровня автопереговоров, который позволяет двум взаимодействующим портам автоматически выбрать наиболее эффективный режим работы, например полудуплексный или дуплексный (этот подуровень является факультативным).

Fast Ethernet поддерживает *три* варианта физической среды:

- волоконно-оптический многомодовый кабель (два волокна);
- витая пара категории 5 (две пары);
- витая пара категории 3 (четыре пары).

Коаксиальный кабель, давший миру первую сеть Ethernet, в число разрешенных сред передачи данных технологии Fast Ethernet не попал. Это общая тенденция многих новых технологий, поскольку на небольших расстояниях витая пара категории 5 позволяет передавать данные с той же скоростью, что и коаксиальный кабель, но сеть получается более дешевой и удобной в эксплуатации. На больших расстояниях оптическое волокно обладает гораздо более широкой полосой пропускания, чем коаксиал, а стоимость сети получается ненамного выше, особенно если учесть высокие затраты на поиск и устранение неисправностей в крупной кабельной коаксиальной системе.

Официальный стандарт 802.3 установил три различные спецификации для физического уровня Fast Ethernet и дал им следующие названия:

- **100Base-TX** — для двухпарного кабеля на неэкранированной витой паре UTP категории 5;
- **100Base-T4** — для четырехпарного кабеля на неэкранированной витой паре UTP категории 3, 4 или 5;
- **100Base-FX** — для многомодового оптоволоконного кабеля с двумя волокнами.

Для всех трех стандартов справедливы перечисленные далее утверждения и характеристики.

> Форматы кадров технологии Fast Ethernet не отличаются от форматов кадров технологий Ethernet 10 Мбит/с.

Межкадровый интервал равен 0,96 мкс, а битовый интервал — 10 нс. Все временные параметры алгоритма доступа (интервал отсрочки, время передачи кадра минимальной длины и т. п.), измеренные в битовых интервалах, остались прежними.

Признаком свободного состояния среды является передача по ней символа простоя источника — соответствующего избыточного кода (а не отсутствие сигналов, как в стандартах Ethernet 10 Мбит/с).

Версия 100Base-T4 носила промежуточный характер, так как она позволяла повысить скорость классического варианта Ethernet в 10 раз, не меняя кабельную систему здания. Так как большинство предприятий и организаций достаточно быстро заменили кабели категории 3 кабелями категории 5, то необходимость в версии 100Base-T4 отпала и оборудование с таким портами перестало выпускаться. Поэтому далее мы рассмотрим детали только спецификаций 100Base-FX и 100Base-TX.

Спецификация 100Base-FX определяет работу протокола Fast Ethernet по многомодовому оптоволокну в полудуплексном и дуплексном режимах. Подуровень кодирования 100Base-FX преобразует байты, получаемые от уровня MAC, в избыточные коды 4B/5B (мы рассматривали этот код в главе 8). В качестве физического метода кодирования используется код IMDD («свет-темнота»).

Избыточный код 4B/5B ко времени разработки технологии Fast Ethernet уже показал свою эффективность в сетях FDDI, поэтому он без изменений был перенесен в спецификацию 100Base-FX/TX. Напомним, что в этом методе каждые четыре бита данных подуровня MAC (называемые символами) представляются пятью битами. Избыточный бит позволяет применить потенциальные коды при представлении каждого из пяти битов в виде электрических или оптических импульсов.

Существование запрещенных комбинаций символов позволяет отбраковывать ошибочные символы, что повышает устойчивость работы сетей 100Base-FX/TX. Так, в Fast Ethernet признаком того, что среда свободна, стала повторяющаяся передача одного из запрещенных для кодирования пользовательских данных символа, а именно символа простоя источника *Idle* (11111). Такой способ позволяет приемнику всегда находиться в синхронизме с передатчиком.

Для отделения кадра Ethernet от символов простоя источника используется комбинация символов начального ограничителя кадра — пара символов *J* (11000) и *K* (10001) кода 4B/5B, а после завершения кадра перед первым символом простоя источника вставляется символ *T* (рис. 12.12).

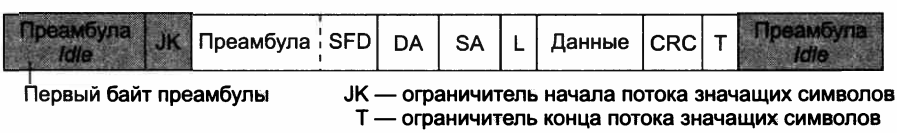


Рис. 12.12. Непрерывный поток данных спецификаций 100Base-FX/TX

После преобразования 4-битных порций кодов MAC в 5-битные порции физического уровня их необходимо представить в виде оптических или электрических сигналов в кабеле, соединяющем узлы сети.

В спецификации 100Base-TX в качестве среды передачи данных используется витая пара UTP категории 5. Как и в спецификации 100Base-FX, здесь применяется избыточный код 4B/5B, а для физического кодирования — код MLT-3 с тремя состояниями электрического сигнала (один из вариантов кода NRZ).

Основным отличием от спецификации 100Base-FX является наличие схемы автопереговоров для выбора режима работы порта.

Схема автопереговоров позволяет двум физически соединенным устройствам, которые поддерживают несколько стандартов физического уровня, отличающихся битовой скоростью и количеством витых пар, согласовать наиболее выгодный режим работы. Обычно процедура автопереговоров происходит при подсоединении сетевого адаптера, который может работать на скоростях 10 и 100 Мбит/с, к концентратору или коммутатору.

Всего в настоящее время определено пять различных режимов работы, которые могут поддерживать устройства 100Base-TX/T4 на витых парах:

- 10Base-T;
- дуплексный режим 10Base-T;
- 100Base-TX;
- 100Base-T4;
- дуплексный режим 100Base-TX.

Режим 10Base-T имеет самый низкий приоритет в переговорном процессе, а дуплексный режим 100Base-TX — самый высокий.

Переговорный процесс происходит при включении питания устройства, а также может быть инициирован в любой момент модулем управления устройства. Устройство, начавшее процесс автопереговоров, посылает своему партнеру пачку специальных импульсов **FLP** (Fast Link Pulse), в которой содержится 8-битное слово, кодирующее предлагаемый режим взаимодействия, начиная с самого приоритетного, поддерживаемого данным узлом. Импульсы FLP имеют длительность 100 нс, как и импульсы LIT, используемые для тестирования целостности физического соединения в стандарте 10Base-T, однако вместо передачи одного импульса LIT через каждые 16 мс здесь через тот же интервал передается пачка импульсов FLP.

Если узел-партнер имеет функцию автопереговоров и также способен поддерживать предложенный режим, он отвечает пачкой импульсов FLP, в которой подтверждает этот режим, и на этом переговоры заканчиваются. Если же узел-партнер не может поддерживать запрошенный режим, то он указывает в своем ответе имеющийся в его распоряжении следующий по степени приоритетности режим и этот режим выбирается в качестве рабочего.

Характеристики производительности Fast Ethernet определяются аналогично характеристикам версии Ethernet 10 Мбит/с с учетом неизменного формата кадра, умножения на 10 битовой скорости (в 10 раз больше) и межкадрового интервала (в 10 раз меньше). В результате получаем:

- максимальная скорость протокола в кадрах в секунду (для кадров минимальной длины с полем данных 46 байт) составляет 148 800;

- полезная пропускная способность для кадров минимальной длины равна 54,8 Мбит/с;
- полезная пропускная способность для кадров максимальной длины (поле данных 1500 байт) равна 97,6 Мбит/с.

Gigabit Ethernet

Достаточно быстро после появления на рынке продуктов Fast Ethernet сетевые интеграторы и администраторы при построении корпоративных сетей почувствовали определенные ограничения. Во многих случаях серверы, подключенные по 100-мегабитному каналу, перегружали магистрали сетей, также работающие на скорости 100 Мбит/с, — магистрали FDDI и Fast Ethernet. Ощущалась потребность в следующем уровне иерархии скоростей. Стандарт Ethernet с битовой скоростью 1000 Мбит/с, получивший название Gigabit Ethernet, был принят в 1998 году.

Проблемы совместимости

Основная идея разработчиков стандарта Gigabit Ethernet состояла в максимальном сохранении идей классической технологии Ethernet при достижении битовой скорости в 1000 Мбит/с.

В результате дебатов были приняты следующие решения:

- сохраняются все форматы кадров Ethernet;
- по-прежнему существует полудуплексная версия протокола, поддерживающая метод доступа CSMA/CD;
- поддерживаются все основные виды кабелей, используемых в Ethernet и Fast Ethernet, в том числе волоконно-оптический кабель, витая пара категории 5, экранированная витая пара.

Несмотря на то что в Gigabit Ethernet не стали встраивать новые функции, поддержание даже достаточно простых функций классического стандарта Ethernet на скорости 1 Гбит/с потребовало решения нескольких сложных задач.

- *Обеспечение приемлемого диаметра сети для работы на разделяемой среде.* В связи с ограничениями, накладываемыми методом CSMA/CD на длину кабеля, версия Gigabit Ethernet для разделяемой среды допускала бы длину сегмента всего в 25 м при сохранении размера кадров и всех параметров метода CSMA/CD неизменными. Так как существует большое количество применений, требующих диаметра сети 100 м, необходимо было каким-то образом решить эту задачу за счет минимальных изменений в технологии Fast Ethernet.
- *Достижение битовой скорости 1000 Мбит/с на оптическом кабеле.* Технология Fibre Channel, физический уровень которой был взят за основу оптоволоконной версии Gigabit Ethernet, обеспечивала скорость передачи данных всего в 800 Мбит/с.
- *Использование в качестве кабеля витой пары.* Такая задача на первый взгляд кажется неразрешимой — ведь даже для 100-мегабитных протоколов требуются достаточно сложные методы кодирования, чтобы уложить спектр сигнала в полосу пропускания кабеля.

Для решения этих задач разработчикам технологии Gigabit Ethernet пришлось внести изменения не только в физический уровень, как это было в случае Fast Ethernet, но и в уровень MAC.

Средства обеспечения диаметра сети в 200 м на разделяемой среде

Для расширения максимального диаметра сети Gigabit Ethernet до 200 м в полудуплексном режиме разработчики технологии предприняли достаточно естественные меры, в основе которых лежало известное соотношение времени передачи кадра минимальной длины и времени оборота (PDV).

Минимальный размер кадра был увеличен (без учета преамбулы) с 64 до 512 байт, или до 4096 бит. Соответственно время оборота увеличилось до 4095 битовых интервалов, что при использовании одного повторителя сделало допустимым диаметр сети около 200 м.

Для увеличения длины кадра до величины, требуемой в новой технологии, сетевой адаптер должен дополнить поле данных до длины 448 байт так называемым **расширением**, представляющим собой поле, заполненное нулями. Формально минимальный размер кадра не изменился, он по-прежнему равняется 64 байт, или 512 бит, и объясняется это тем, что поле расширения помещается после поля контрольной суммы кадра (FCS). Соответственно значение этого поля не включается в контрольную сумму и не учитывается при указании длины поля данных в поле длины. Поле расширения является просто расширением сигнала несущей частоты, необходимым для корректного обнаружения коллизий.

Для сокращения накладных расходов в случае использования слишком длинных кадров при передаче коротких квитанций разработчики стандарта разрешили конечным узлам *передавать несколько кадров подряд без возвращения среды* другим станциям. Такой режим получил название **режима пульсаций**. Станция может передать подряд несколько кадров с общей длиной не более 65 536 бит, или 8192 байт. При передаче нескольких небольших кадров станции можно не дополнять первый кадр до размера в 512 байт за счет поля расширения, а передавать несколько кадров подряд до исчерпания предела в 8192 байт (в этот предел входят все байты кадра, в том числе преамбула, заголовок, данные и контрольная сумма). Предел 8192 байт называется **длиной пульсации**. Если предел длины пульсации достигается в середине кадра, то кадр разрешается передать до конца. Увеличение «совмещенного» кадра до 8192 байт несколько задерживает доступ к разделяемой среде других станций, но при скорости 1000 Мбит/с эта задержка не столь существенна.

Спецификации физической среды стандарта Gigabit Ethernet

Для поддержания различных физических сред физический уровень Gigabit Ethernet имеет такую же модульную структуру, как и физический уровень Fast Ethernet, с тем отличием, что вместо интерфейса MII в нем применяется интерфейс GMII (Gigabit MII), работающий на скорости 1 Гбит/с.

В стандарте 802.3z определены следующие типы физической среды:

- одномодовый волоконно-оптический кабель;
- многомодовый волоконно-оптический кабель 62,5/125;
- многомодовый волоконно-оптический кабель 50/125;
- экранированный сбалансированный медный кабель.

Для передачи данных по многомодовому волоконно-оптическому кабелю стандарт предписывает применение излучателей, работающих на двух длинах волн: 850 и 1300 нм. Применение светодиодов с длиной волны 850 нм объясняется тем, что они намного дешевле, чем светодиоды, работающие на волне 1300 нм, хотя при этом максимальная длина кабеля уменьшается, так как затухание многомодового оптоволокна на волне 850 нм более чем в два раза выше, чем на волне 1300 нм.

Для многомодового оптоволокна стандарт Gigabit Ethernet определяет спецификации 1000Base-SX и 1000Base-LX. В первом случае используется длина волны 850 нм (S означает Short Wavelength — короткая длина волны), а во втором — 1300 нм (L означает Long Wavelength — длинная длина волны). Спецификация 1000Base-SX разрешает использовать только многомодовый кабель, при этом его максимальная длина составляет около 500 м.

Для спецификации 1000Base-LX в качестве источника излучения всегда применяется полупроводниковый лазерный диод с длиной волны 1300 нм. Спецификация 1000Base-LX позволяет работать как с многомодовым (максимальное расстояние до 500 м), так и с одномодовым кабелем (максимальное расстояние зависит от мощности передатчика и качества кабеля и может доходить до нескольких десятков километров).

В спецификациях 1000Base-SX и 1000Base-LX подуровень кодирования преобразует байты уровня MAC в коды 8B/10B (а не 4B/5B, как в стандарте Fast Ethernet).

Gigabit Ethernet на витой паре категории 5

Как известно, каждая пара кабеля категории 5 имеет гарантированную полосу пропускания до 100 МГц. Организовать передачу по такому кабелю данных со скоростью 1000 Мбит/с очень трудно, так как применяемые в локальных сетях методы кодирования имеют на такой тактовой частоте спектр с шириной, намного превышающий 100 МГц. В качестве решения было предложено организовать параллельную передачу данных одновременно по всем четырем парам кабеля.

Это сразу снизило скорость передачи данных по каждой паре до 250 Мбит/с. Однако и для такой скорости необходимо было придумать метод кодирования со спектром, не превышающим 100 МГц. Усложняло задачу и то обстоятельство, что стандарт Gigabit Ethernet должен поддерживать не только полудуплексный, но и дуплексный режим. На первый взгляд кажется, что одновременное использование четырех пар лишает сеть возможности работы в дуплексном режиме, так как не остается свободных пар для одновременной передачи данных в двух направлениях — от узла и к узлу.

Тем не менее проблемная группа 802.3ab нашла решения обеих проблем.

Для физического кодирования данных был применен код PAM5 с пятью уровнями потенциала: –2, –1, 0, +1, +2. В этом случае за один такт по одной паре передается 2,322 бит информации ($\log_2 5$). Следовательно, для достижения скорости 250 Мбит/с тактовую частоту 250 МГц можно уменьшить в 2,322 раза. Разработчики стандарта решили использовать несколько более высокую частоту, а именно 125 МГц. При этой тактовой частоте код PAM5 имеет спектр у́же, чем 100 МГц, то есть он может быть передан без искажений по кабелю категории 5.

В каждом такте передается не $2{,}322 \times 4 = 9{,}288$ бит информации, а 8. Это и дает искомую суммарную скорость 1000 Мбит/с. Передача ровно восьми битов в каждом такте дости-

гается за счет того, что подуровень кодирования PCS преобразует байты, получаемые от уровня MAC через интерфейс GMII, в логический код 4D-PAM5, который состоит из четырех символов, каждый их которых имеет пять состояний (пять состояний символа соответствует пяти состояниям физического кода PAM5).

При кодировании информации используются не все 625 ($5^4 = 625$) комбинаций кода PAM5, а только 256 ($2^8 = 256$). Оставшиеся комбинации приемник задействует для контроля принимаемой информации и выделения правильных комбинаций на фоне шума.

Для организации дуплексного режима разработчики спецификации 802.3ab применили технику выделения принимаемого сигнала из суммарного. Два передатчика работают навстречу друг другу по каждой из четырех пар в одном и том же диапазоне частот.

Для отделения принимаемого сигнала от собственного приемник вычитает из результирующего сигнала известный ему свой сигнал. Естественно, что это не простая операция и для ее выполнения используются специальные процессоры цифровой обработки сигнала (Digital Signal Processor, DSP).

Вариант технологии Gigabit Ethernet на витой паре расширил *процедуру автопереговоров*, введенную стандартом 100Base-T, за счет включения туда дуплексного и полудуплексного режимов работы на скорости 1000 Мбит/с. Поэтому порты многих коммутаторов Ethernet на витой паре являются универсальными в том смысле, что могут работать на любой из трех скоростей (10, 100 или 1000 Мбит/с).

Характеристики производительности Gigabit Ethernet зависят от того, использует ли коммутатор режим передачи кадров с расширением или же передает их в режиме пульсаций.

В режиме пульсаций на периоде пульсации мы получаем характеристики, в 10 раз отличающиеся от характеристик Fast Etherhet:

- максимальная скорость протокола в кадрах в секунду (для кадров минимальной длины с полем данных 46 байт) составляет 1 488 000;
- полезная пропускная способность для кадров минимальной длины равна 548 Мбит/с;
- полезная пропускная способность для кадров максимальной длины (поле данных 1500 байт) равна 976 Мбит/с.

10G Ethernet

Стандарт **10G Ethernet** определяет только дуплексный режим работы, поэтому он используется исключительно в *коммутируемых* локальных сетях.

Формально этот стандарт имеет обозначение **IEEE 802.3ae** и является дополнением к основному тексту стандарта 802.3. Формат кадра остался неизменным, при этом расширение кадра, введенное в стандарте Gigabit Ethernet, не используется, так как нет необходимости обеспечивать распознавание коллизий.

Стандарт 802.3ae, принятый в 2002 году, описывает несколько новых спецификаций физического уровня, которые взаимодействуют с уровнем MAC через новый интерфейс **XGMII** (eXtended Gigabit Medium Independent Interface — расширенный интерфейс независимого доступа к гигабитной среде). Интерфейс XGMII предусматривает параллельный обмен четырьмя байтами, образующими четыре потока данных. Как мы видим, идея распаралле-

ливания потоков данных, хорошо зарекомендовавшая себя в предыдущих версиях Ethernet как средство снижения требований к полосе пропускания отдельного канала, здесь нашла отражение в структуре интерфейса между уровнем MAC и физическим уровнем. Наличие четырех независимых потоков на входе физического уровня упрощает организацию параллельных потоков данных в приемопередатчиках Ethernet.

Стандарт 10G Ethernet определяет три группы физических интерфейсов:

- 10GBase-X;
- 10Gbase-R4;
- 10GBase-W.

Они отличаются способом логического кодирования данных: в варианте 10Base-X применяется код 8B/10B, в остальных двух — код 64B/66B. Все они для передачи данных используют оптическую среду.

Группа 10GBase-X в настоящее время состоит из одного интерфейса подуровня PMD — 10GBase-LX4. Буква L говорит о том, что информация передается с помощью волн второго диапазона прозрачности, то есть 1310 нм. Информация в каждом направлении передается одновременно с помощью четырех волн (что отражает цифра 4 в названии интерфейса), которые мультиплексируются на основе техники WDM. Каждый из четырех потоков интерфейса XGMII передается в оптическом волокне со скоростью 2,5 Гбит/с.

Максимальное расстояние между передатчиком и приемником стандарта 10GBase-LX4 на многомодовом волокне равно 200–300 м (в зависимости от полосы пропускания волокна), на одномодовом — 10 км.

В каждой из групп **10GBase-R** и **10GBase-W** может быть три варианта подуровня PMD: S, L и E в зависимости от используемого для передачи информации диапазона волн — 850, 1310 или 1550 нм соответственно. Таким образом, существуют интерфейсы 10GBase-SR, 10GBase-LR и 10GBase-ER, а также 10GBase-SW, 10GBase-LW, 10GBase-EW. Каждый из них передает информацию с помощью одной волны соответствующего диапазона.

Спецификации 10GBase-R обеспечивают эффективную скорость передачи данных в 10 Гбит/с, для этого битовая скорость оборудования равна 10,3125 Гбит/с (увеличение битовой скорости необходимо для компенсации избыточности кода 64B/66B).

В отличие от 10GBase-R физические интерфейсы группы 10GBase-W обеспечивают скорость передачи и формат данных, совместимые с интерфейсом OTN ODU-2. Пропускная способность интерфейсов группы W равна 9,95328 Гбит/с, а эффективная скорость передачи данных — 9,58464 Гбит/с (часть пропускной способности тратится на заголовки кадров OTN). Из-за того что скорость передачи информации у этой группы интерфейсов ниже, чем 10 Гбит/с, они могут взаимодействовать только между собой, то есть соединение, например, интерфейсов 10GBase-LR и 10Base-LW невозможно.

Физические интерфейсы, работающие в окне прозрачности E, обеспечивают передачу данных на расстояния до 40 км. Это позволяет строить не только локальные сети, но и сети мегаполисов, что нашло отражение в поправках к исходному тексту стандарта 802.3.

В 2006 году была принята спецификация **10GBase-T**, которая дает возможность использовать знакомые администраторам локальных сетей кабели на витой паре. Правда, обязательным требованием является применение кабелей категории 6 или 6a: в первом случае максимальная длина кабеля не должна превышать 55 м, во втором — 100 м.

Стандарт 10G Ethernet развивается за счет пополнения его семейства физических интерфейсов новыми спецификациями, например спецификацией 10GBase-KX4, предназначенной для работы по четырем проводникам шасси сетевых устройств.

100G и 40G Ethernet

Хотя 10 Гбит/с и является довольно высокой скоростью передачи данных, достаточной для многих приложений, например для телевидения высокого разрешения, но и она по прошествии нескольких лет перестала удовлетворять потребности постоянно растущего трафика Интернета и новых приложений.

Поэтому в 2006 году рабочая группа IEEE 802.3 образовала группу по изучению высокоскоростного варианта Ethernet (High Speed Study Group, HSSG). Анализ ситуации показал, что целесообразно стандартизировать две новые скорости Ethernet — 40 и 100 Гбит/с. Первая скорость предназначалась для серверов, вторая — для интерфейсов коммутаторов и маршрутизаторов, работающих на магистралях сетей и агрегирующих потоки данных многих приложений. В результате в 2008 году была создана целевая группа 802.3ba, которая и предложила соответствующий вариант стандарта Ethernet, впервые описывающего упомянутые две скорости передачи данных. Этот стандарт вошел в общий стандарт 802.3-2012 как одна из частей (наряду с частями, описывающими остальные скорости Ethernet).

Архитектура стандарта 802.3ba обобщает подход, использованный в технологии 10G Ethernet, а именно распараллеливание общего потока данных от уровня MAC на несколько потоков. Этот подход позволяет снизить битовую скорость каждого из параллельных потоков, тем самым упрощая реализацию высокоскоростного приемопередатчика.

В стандарте 802.3ba распараллеливание применяется на двух этапах передачи данных от уровня MAC к уровню физического интерфейса (Physical Media Dependence, PMD). Сначала уровень согласования распараллеливает общий последовательный поток данных, поступающий от уровня MAC, на восемь потоков, которые параллельно поступают на подуровень физического кодирования PCS через интерфейс **XLGMII** (для скорости 40 Гбит/с, буквы XL и означают римское число 40) или интерфейс **CGMII** (для скорости 100 Гбит/с, от C — римское число 100).

Подуровень PCS выполняет кодирование данных, поступающих по восьми потокам, в соответствии с кодировкой 64B/66B (она одна для всех вариантов физического интерфейса), а затем направляет их четырьмя (для скорости 40 Гбит/с) или двадцатью (для скорости 100 Гбит/с) потоками на подуровни PMA и PMD, которые реализуются, как правило, отдельным модулем — приемопередатчиком (трансивером). В подуровнях PMA/PMD эти потоки могут группироваться в один или несколько каналов, передаваемых отдельными волнами (если применяется мультиплексирование WDM) или отдельными медными проводниками.

Выбор 20 потоков не случаен, он обеспечивает высокую гибкость для спецификаций физической среды, так как дает возможность сформировать 1, 2, 4, 5, 10 или 20 независимых физических каналов.

Рисунок 12.13 иллюстрирует работу подуровня PCS по распараллеливанию данных по потокам.

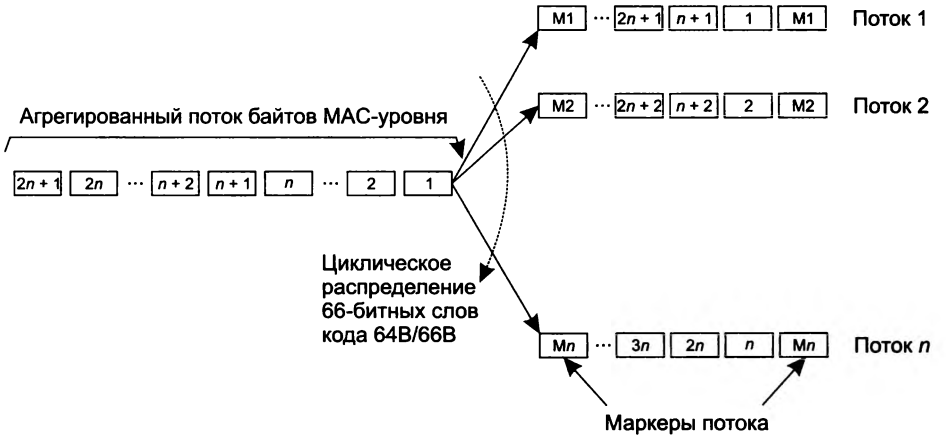


Рис. 12.13. Распределение данных подуровнем PCS по потокам

66-битные слова кода 64B/66B циклически распределяются между потоками данных. После каждых 16 384 слов в поток помещается специальный код маркера потока, который помогает подуровню PCS приемника правильно демультиплексировать потоки в общий поток. Для сохранения синхронности потока для вставки кода маркера из потока периодически удаляются коды межкадрового интервала (IPG).

Для 100 Gigabit Ethernet рабочей группой IEEE 802.3 разработаны различные стандарты физического уровня (табл. 12.1). По назначению они делятся на стандарты, предназначенные для работы в пределах шасси одного устройства (стандарты KR), для соединения устройств в пределах одной или нескольких стоек (CR и T), одного здания (SR и FR) или же для создания глобальных соединений между различными центрами данных (LR и ER). Почти все варианты физического уровня 100GE обеспечивают необходимую суммарную битовую скорость за счет использования нескольких параллельных потоков данных — как видно из таблицы, четырех или десяти. Эти параллельные потоки образуются либо отдельными проводниками (печатными проводниками для вариантов KR, витыми парами для варианта T, парами твинаксиального кабеля для вариантов CR или же парами оптических волокон для варианта SR), либо отдельными волнами технологии WDM в вариантах LR и ER.

Таблица 12.1. Стандарты физического уровня для технологии 100 Gigabit Ethernet

Гарантированное расстояние и тип среды	40 Gigabit Ethernet	100 Gigabit Ethernet
>1 м шасси	40GBase-KR4	
>7 м твинаксиальный медный кабель	40GBase-CR4	100GBase-CR10
>100 м OM3* MMF	40GBase-SR4	100GBase-SR10
>150 м OM4 MMF	40GBase-SR4	100GBase-SR10
>10 км SMF	40GBase-LR4	100GBase-LR10
>40 км SMF		100GBase-ER10

Гарантированное расстояние и тип среды	40 Gigabit Ethernet	100 Gigabit Ethernet
>2 км SMF	40GBase-FR	
>30 м витой пары категории 8 (4 пары)	40GBase-T*	

* OM3 и OM4 — типы многомодового оптического волокна (оптимизированного), отличающиеся характеристиками передачи сигнала волны 850 нм.

** На момент написания книги стандарт находился в разработке.

Архитектура коммутаторов

Для ускорения операций коммутации сегодня во всех коммутаторах используются заказные специализированные БИС — ASIC, которые оптимизированы для выполнения основных операций коммутации. Часто в одном коммутаторе имеется несколько специализированных БИС, каждая из которых выполняет функционально законченную часть операций.

Важную роль в построении коммутаторов играют также программируемые микросхемы **FPGA** (Field-Programmable Gate Array — программируемый в условиях эксплуатации массив вентилей). Эти микросхемы могут выполнять все функции, которые выполняют микросхемы ASIC, но в отличие от последних могут программироваться и перепрограммироваться производителями коммутаторов (и даже пользователями). Это свойство позволило резко удешевить процессоры портов коммутаторов, выполняющих сложные операции, например профилирование трафика, так как производитель FPGA выпускает свои микросхемы массово, а не по заказу того или иного производителя оборудования. Кроме того, применение микросхем FPGA позволяет производителям коммутаторов оперативно вносить изменения в логику работы порта при появлении новых стандартов или изменении действующих.

Помимо процессорных микросхем для успешной неблокирующей работы коммутатору нужно иметь быстродействующий *узел обмена*, предназначенный для передачи кадров между процессорными микросхемами портов.

В настоящее время в коммутаторах узел обмена строится на основе одной из трех схем:

- коммутационная матрица;
- общая шина;
- разделяемая многовходовая память.

Часто эти три схемы комбинируются в одном коммутаторе.

Коммутационная матрица обеспечивает наиболее простой способ взаимодействия процессоров портов, и именно этот способ был реализован в первом промышленном коммутаторе локальных сетей. Однако реализация матрицы возможна только для определенного числа портов, причем сложность схемы возрастает пропорционально квадрату количества портов коммутатора (рис. 12.14).

Более детальное представление одного из возможных вариантов реализации коммутационной матрицы для восьми портов дано на рис. 12.15. Входные блоки процессоров портов на основании просмотра адресной таблицы коммутатора определяют по адресу назначения номер выходного порта. Эту информацию они добавляют к байтам исходного кадра в виде специального ярлыка — тега. Для данного примера тег представляет собой просто трехразрядное двоичное число, соответствующее номеру выходного порта.

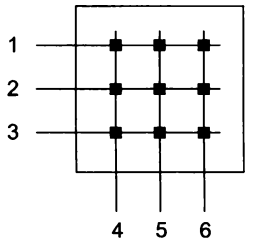


Рис. 12.14. Коммутационная матрица

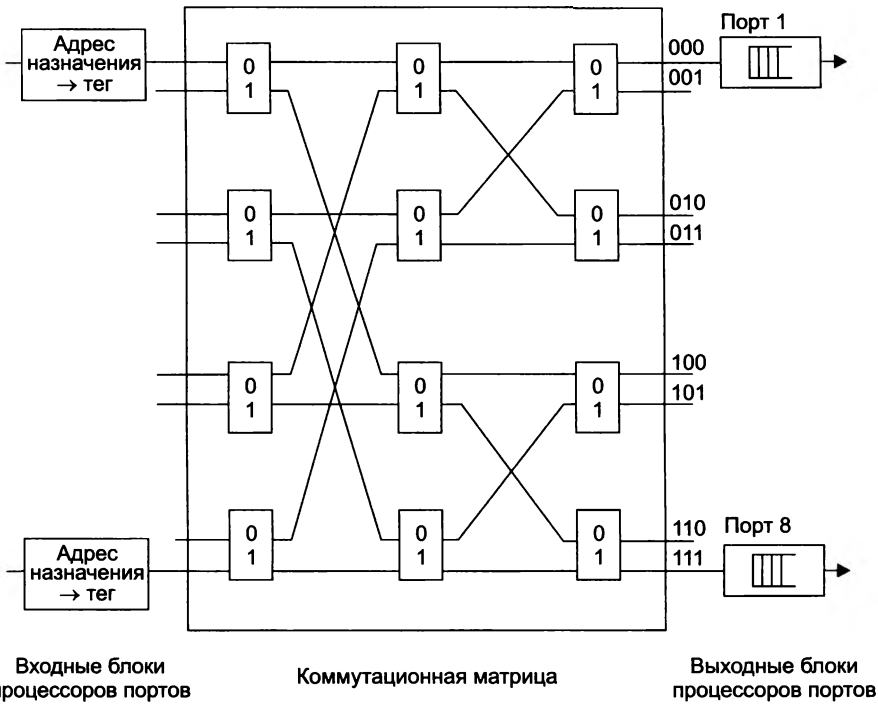


Рис. 12.15. Реализация коммутационной матрицы 8 × 8 с помощью двоичных переключателей

Матрица состоит из трех уровней двоичных переключателей, которые соединяют свой вход с одним из двух выходов в зависимости от значения бита тега. Переключатели первого уровня управляются первым битом тега, второго — вторым, а третьего — третьим.

Матрица может быть реализована и иначе, на основании комбинационных схем другого типа, но ее особенностью все равно остается технология коммутации физических каналов. Известным недостатком этой технологии является отсутствие буферизации данных внутри коммутационной матрицы — если составной канал невозможно построить из-за занятости выходного порта или промежуточного коммутационного элемента, то данные должны накапливаться в их источнике, в данном случае — во входном блоке порта, принявшего кадр. Основные достоинства таких матриц — высокая скорость коммутации и регулярная

структура, которую удобно реализовывать в интегральных микросхемах. Зато после реализации матрицы $N \times N$ в составе БИС проявляется еще один ее недостаток — сложность наращивания числа коммутируемых портов.

В **коммутаторах с общей шиной** процессоры портов связывают высокоскоростной шиной, используемой в режиме разделения времени.

Пример такой архитектуры приведен на рис. 12.16. Чтобы шина не блокировала работу коммутатора, ее производительность должна равняться по крайней мере сумме производительностей всех портов коммутатора. Для модульных коммутаторов характерно то, что путем удачного подбора модулей с низкоскоростными портами можно обеспечить неблокирующий режим работы, но в то же время некоторые сочетания модулей с высокоскоростными портами могут приводить к структурам, у которых узким местом является общая шина.

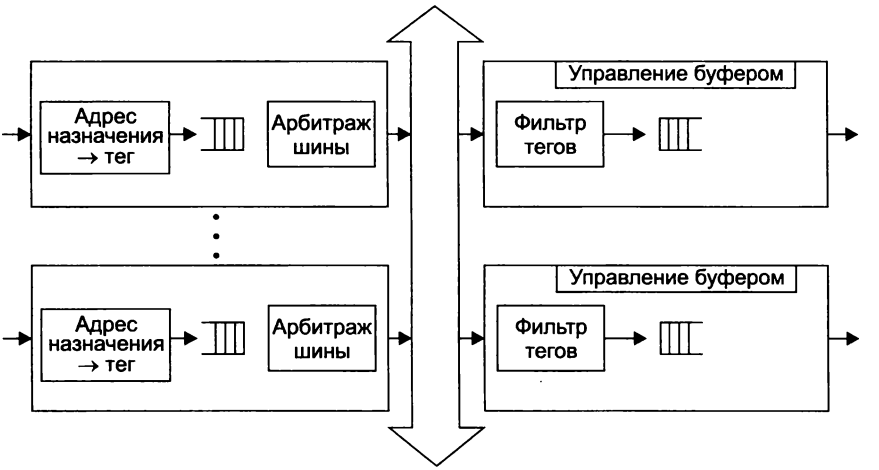


Рис. 12.16. Архитектура коммутатора с общей шиной

Кадр должен передаваться по шине небольшими частями, по несколько байтов, чтобы передача кадров между портами происходила в псевдопараллельном режиме, не внося задержек в передачу кадра в целом. Размер такой ячейки данных определяется производителем коммутатора. Некоторые производители выбирают в качестве порции данных, переносимых по шине за одну операцию, ячейку ATM с ее полем данных в 48 байт. Такой подход облегчает трансляцию протоколов локальных сетей в протокол ATM, если коммутатор поддерживает эти технологии. Кроме того, небольшой размер ячейки (ее формат может быть и фирменным, так как перенос данных между портами является сугубо внутренней операцией) уменьшает задержки доступа порта к общей шине.

Входной блок процессора помещает в ячейку, переносимую по шине, тег, в котором указывает номер порта назначения. Каждый выходной блок процессора порта содержит фильтр тегов, который выбирает теги, предназначенные данному порту.

Шина, так же как и коммутационная матрица, не может осуществлять промежуточную буферизацию, но поскольку данные кадра разбиваются на небольшие ячейки, задержек с начальным ожиданием доступности выходного порта в такой схеме нет — здесь работает принцип коммутации пакетов, а не каналов.

Разделяемая многовходовая память представляет собой третью базовую архитектуру взаимодействия портов. Пример такой архитектуры приведен на рис. 12.17.

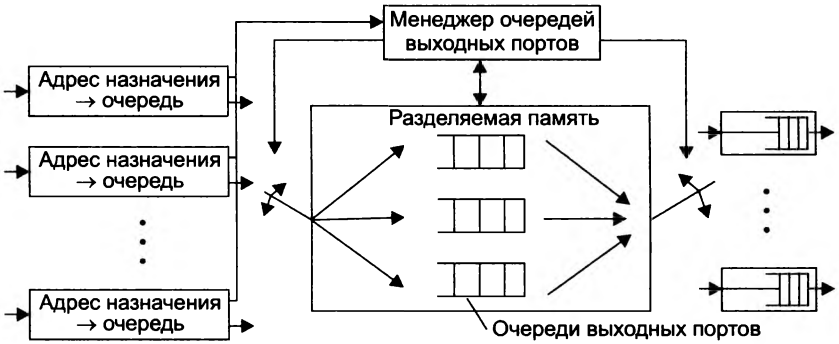


Рис. 12.17. Архитектура коммутаторов с разделяемой памятью

Входные блоки процессоров портов соединяются с переключаемым входом разделяемой памяти, а выходные блоки этих же процессоров — с ее переключаемым выходом. Переключением входа и выхода разделяемой памяти управляет *менеджер очередей выходных портов*. В разделяемой памяти менеджер организует несколько очередей данных, по одной для каждого выходного порта. Входные блоки процессоров передают менеджеру портов запросы на запись данных в очередь того порта, который соответствует адресу назначения кадра. Менеджер по очереди подключает вход памяти к одному из входных блоков процессоров, и тот переписывает часть данных кадра в очередь определенного выходного порта. По мере заполнения очередей менеджер производит также поочередное подключение выхода разделяемой памяти к выходным блокам процессоров портов и данные из очереди переписываются в выходной буфер процессора.

Применение общей буферной памяти, гибко распределяемой менеджером между отдельными портами, снижает требования к размеру буферной памяти процессора порта. Однако буферная память должна быть достаточно быстродействующей для поддержания необходимой скорости обмена данными между N портами коммутатора.

У каждой из описанных архитектур есть свои достоинства и недостатки, поэтому часто в сложных коммутаторах эти архитектуры применяются в комбинации друг с другом. Пример такого комбинирования приведен на рис. 12.18.

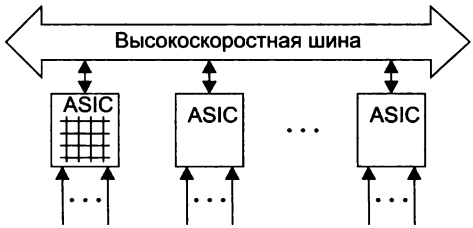


Рис. 12.18. Комбинирование архитектур коммутационной матрицы и общей шины

Комбинированный коммутатор состоит из модулей с фиксированным количеством портов (2–12), выполненных на основе специализированной БИС, реализующей архитектуру коммутационной матрицы. Если порты, между которыми нужно передать кадр данных, принадлежат одному модулю, то передача кадра осуществляется процессорами модуля на основе имеющейся в модуле коммутационной матрицы. Если же порты принадлежат разным модулям, то процессоры общаются по общей шине. В такой архитектуре передача кадров внутри модуля происходит быстрее, чем при межмодульной передаче, так как коммутационная матрица — это наиболее быстрое, хотя и наименее масштабируемое средство взаимодействия портов. Скорость внутренней шины коммутаторов может достигать нескольких гигабит в секунду, а у наиболее мощных моделей — до нескольких десятков Гбит/с.

(S) *Конструктивное исполнение коммутаторов*

Выводы

Для логической структуризации сети применяются мосты и их современные преемники — коммутаторы локальных сетей. Устройства обоих типов работают на основе одного и того же стандарта IEEE 802.1D, но коммутаторы обладают гораздо более высоким быстродействием за счет параллельной обработки потоков данных.

Коммутаторы являются самообучающимися устройствами, так как строят таблицы продвижения автоматически на основе слежения за передаваемыми кадрами.

Недостатком коммутаторов является невозможность работы в сетях с петлевидными связями. Другим недостатком сетей, построенных на коммутаторах, является отсутствие защиты от широковещательного шторма.

Применение коммутаторов позволяет сетевым адаптерам использовать дуплексный режим работы. В этом режиме отсутствует этап доступа к разделяемой среде, а общая скорость передачи данных удваивается.

В дуплексном режиме для борьбы с перегрузками коммутаторов используется метод обратной связи, описанный в стандарте 802.3x. Он позволяет приостановить на некоторое время поступление кадров от непосредственных соседей перегруженного коммутатора.

Основными характеристиками производительности коммутатора являются: скорость фильтрации кадров, скорость продвижения кадров, общая пропускная способность по всем портам в мегабитах в секунду, задержка передачи кадра.

Потребности в высокоскоростной и в то же время недорогой технологии для подключения к сети мощных рабочих станций привели к созданию нескольких скоростных версий Ethernet: Fast Ethernet со скоростью 100 Мбит/с, Gigabit Ethernet со скоростью 1 Гбит/с, 10G Ethernet со скоростью 10 Гбит/с и 100G Ethernet со скоростью 100 Гбит/с.

Существует несколько основных вариантов внутренней архитектуры коммутаторов, в основе которых лежат:

- коммутационная матрица;
- разделяемая память;
- общая шина.

Кроме того, применяется комбинирование основных вариантов в одном устройстве.

Контрольные вопросы

1. Что из перечисленного можно отнести к недостаткам сетей на разделяемой среде:
   а) ограниченный диаметр сети;
   б) сложность подключения нового узла к сети;
   в) плохая масштабируемость;
   г) сложность организации широковещания.
2. На основе изучения каких адресов автоматически строится таблица продвижения моста? Варианты ответов:
   а) MAC-адресов назначения;
   б) MAC-адресов источника.
3. К каким негативным последствиям приводит наличие петель в сети, построенной на коммутаторах, работающих в соответствии с алгоритмом прозрачного моста? Варианты ответов:
   а) кадры могут дублироваться;
   б) кадры могут зацикливаться;
   в) кадры могут отбрасываться.
4. Совпадают ли форматы кадров 100 Гбит/с Ethernet и 10 Гбит/с?
5. Может ли в технологии 100G Ethernet использоваться разделяемая среда?

ГЛАВА 13 Отказоустойчивость и виртуализация локальных сетей

Алгоритм покрывающего дерева

В коммутируемых локальных сетях проблема обеспечения надежности сети имеет свою специфику: базовый протокол прозрачного моста корректно работает только в сети с *древовидной топологией*, в которой между любыми двумя узлами сети существует единственный маршрут. Тем не менее очевидно, что для надежной работы сети необходимо наличие альтернативных маршрутов между узлами, которые можно было бы использовать при отказе основного маршрута. Наиболее простым решением этой проблемы является построение сети с альтернативными маршрутами, ручное нахождение связной древовидной топологии и ручное блокирование (то есть перевод в административное состояние «отключен») всех портов, которые не входят в найденную топологию. В случае отказа сети этот процесс должен повторяться, опять же в ручном режиме. Понятно, что надежность сети в этом случае оказывается не очень высокой, так как время пребывания ее в неработоспособном состоянии будет исчисляться минутами: сначала нужно обнаружить отказ и локализовать его (то есть не только зафиксировать факт, что в сети что-то перестало работать, но и понять, какая именно связь пострадала и требует обхода), затем найти новый работоспособный вариант топологи сети (если он, конечно, существует), а потом его сконфигурировать.

Для автоматического выполнения перечисленных действий, то есть нахождения и конфигурирования активной древовидной топологии, мониторинга состояния ее связей и перехода к новой древовидной топологии при обнаружении отказа связи в коммутируемых локальных сетях используются **алгоритм покрывающего дерева** (Spanning Tree Algorithm, STA) и реализующий его **протокол покрывающего дерева** (Spanning Tree Protocol, STP).

Алгоритм покрывающего дерева, разработанный достаточно давно, в 1983 году, был признан IEEE удачным решением и включен в ту же спецификацию 802.1D, в которой описывается и сам алгоритм прозрачного моста. Фактически протокол STP является специфической упрощенной версией протокола маршрутизации, упрощение заключается в том, что кадры направляются по активному маршруту независимо от их адреса назначения, в то время как в протоколах маршрутизации активный маршрут выбирается для каждого адреса индивидуально.

Сегодня протокол STP широко применяется в наиболее массовых устройствах современных локальных сетей — коммутаторах. Версия протокола STP, получившая название RSTP (Rapid STP, то есть быстрый протокол покрывающего дерева) работает значительно быстрее, затрачивая на поиск новой топологии несколько секунд.

Протокол STP

Протокол STP формализует сеть (рис. 13.1, *а*) в виде графа (рис. 13.1, *б*), вершинами которого являются коммутаторы и сегменты сети.

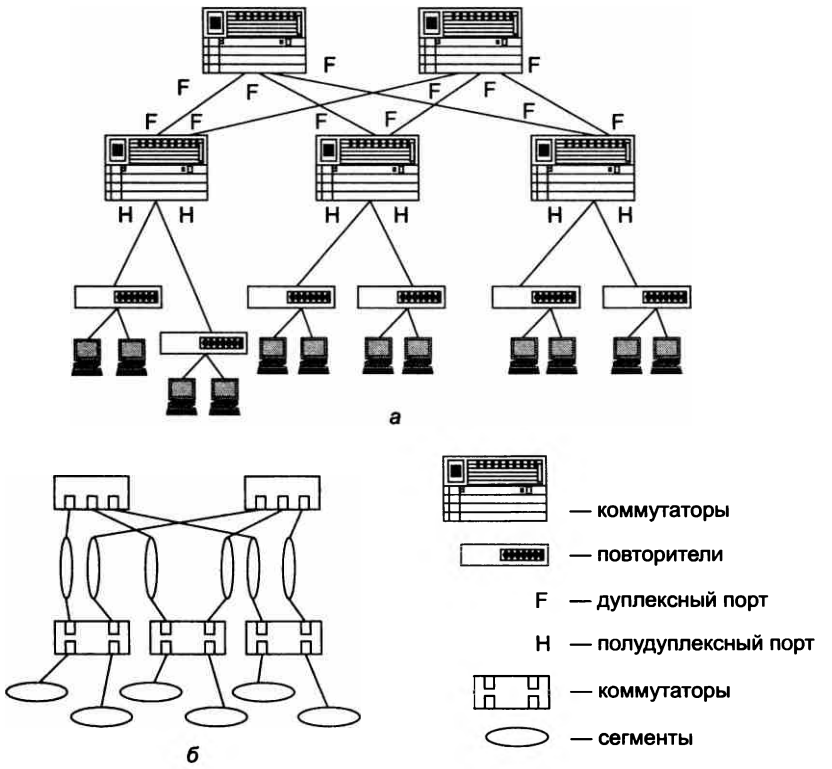


Рис. 13.1. Формализованное представление сети в соответствии с алгоритмом STA

Сегмент — это связная часть сети, не содержащая коммутаторов (и маршрутизаторов). Сегмент может быть разделяемым (во времена создания алгоритма STA это был единственный тип сегмента) и включать устройства физического уровня — повторители/концентраторы, существование которых коммутатор, будучи устройством канального уровня, «не замечает». Сегмент также может представлять собой двухточечный канал; в коммутируемых локальных сетях, применяемых сегодня, это единственный тип сегмента.

Протокол покрывающего дерева обеспечивает построение древовидной топологии связей с единственным путем минимальной длины от каждого коммутатора и от каждого сегмента до некоторого выделенного **корневого коммутатора** — корня дерева. *Единственность* пути гарантирует отсутствие петель, а *минимальность* расстояния — рациональность маршрутов следования трафика от периферии сети к ее магистрали, роль которой исполняет корневой коммутатор.

В качестве расстояния в STA используется **метрика** — традиционная для протоколов маршрутизации величина, обратно пропорциональная пропускной способности сегмента.

Также в STA метрика определяется как *условное время передачи бита сегментом*. В текущей версии стандарта 802.1D-2004 используются такие значения метрик, которые расширяют диапазон скоростей сегментов до 10 Тбит/с (то есть с большим запасом относительно сегодняшнего уровня максимальной для Ethernet скорости в 100 Гбит/с), давая такому сегменту значение 2; соответственно сегмент 100 Гбит/с получает значение 200, 10 Гбит/с — 2000, 1 Гбит/с — 20 000, 100 Мбит/с — 200 000, а 10 Мбит/с — 2 000 000.

Протокольными единицами данных моста (Bridge Protocol Data Unit, BPDU) называются специальные пакеты, которыми периодически обмениваются коммутаторы для автоматического определения конфигурации дерева. Пакеты BPDU переносят данные об идентификаторах коммутаторов и портов, а также о расстоянии до корневого коммутатора. Существуют два типа сообщений, которые переносят пакеты BPDU: конфигурационные сообщения, называемые также сообщениями Hello (с интервалом 2 с), и сообщения с уведомлениями об изменении конфигурации. Для доставки BPDU используется групповой адрес 01:80:C2:00:00:00, позволяющий организовать эффективный обмен данными.

Три этапа построения дерева

На рис. 13.2 приведен пример сети стандарта 802.1D-2004, который иллюстрирует работу протокола STP. Мы также будем использовать этот пример в своем описании.

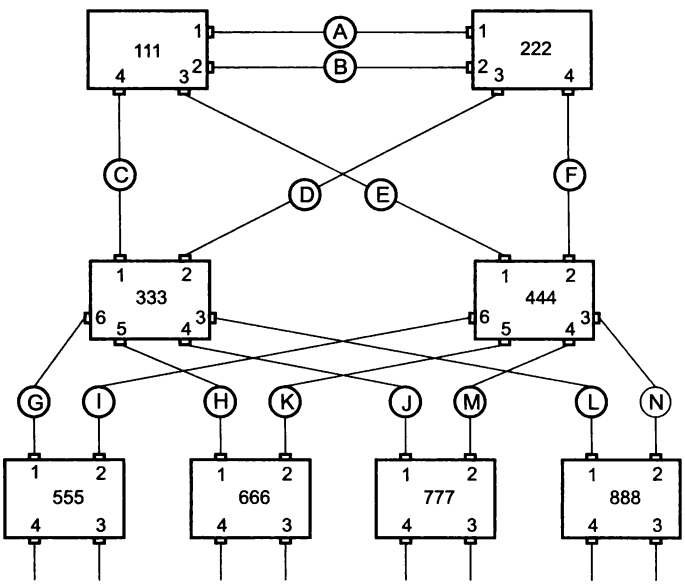


Рис. 13.2. Пример сети, иллюстрирующей работу STP

В этом примере сеть построена на восьми коммутаторах, которые имеют идентификаторы со значениями от 111 до 888. В стандарте в качестве идентификатора коммутатора используется его MAC-адрес, к которому добавляются два старших байта, конфигурируемые вручную, — администратор может использовать их для вмешательства в выбор корневого коммутатора. Для удобства записи на рисунке указаны сокращенные до трех разрядов

значения MAC-адресов коммутаторов. Все коммутаторы соединены друг с другом двухточечными связями, которые образуют сегменты *A*–*N*. Порты 3 и 4 коммутаторов с 555 по 888 соединены с конечными узлами сети, то есть компьютерами (на рисунке не показаны). Все связи в сети — это связи со скоростью 100 Мбит/с (Fast Ethernet).

Алгоритм STA определяет активную конфигурацию сети за три этапа.

1. *Определение корневого коммутатора, от которого строится дерево.*

   В качестве корневого коммутатора выбирается коммутатор с *наименьшим значением идентификатора*. В исходном состоянии каждый коммутатор считает себя корневым, поэтому он генерирует и передает своим соседям сообщения Hello, в которых помещает свой идентификатор в качестве идентификатора корневого коммутатора. Как только коммутатор получает от соседа сообщение Hello, в котором содержится идентификатор корневого коммутатора, меньший его собственного, он перестает считать себя корневым коммутатором и генерировать свои сообщения Hello, но начинает ретранслировать сообщения Hello, получаемые от соседей.

   В нашем примере мы предполагаем, что администратор не стал менять старшие байты коммутаторов, так что у всех коммутаторов они остались равными значению 32 768 (значение, предлагаемое по умолчанию), и корневым коммутатором стал коммутатор с идентификатором 111.

2. *Выбор корневого порта для каждого коммутатора.*

   Корневым портом коммутатора является тот порт, расстояние от которого до корневого коммутатора является минимальным. Сам корневой коммутатор корневых портов не имеет.

   Для определения корневого порта каждый коммутатор использует пакеты Hello, ретранслируемые ему другими коммутаторами. На основании этих пакетов каждый коммутатор определяет минимальные расстояния от всех своих портов до корневого коммутатора и выбирает порт с наименьшим значением в качестве корневого. При равенстве расстояний выбирается порт с наименьшим значением идентификатора порта (это порядковый номер порта в коммутаторе).

   Например, у коммутатора 222 порты 1 и 2 находятся на одинаковом расстоянии до корневого коммутатора 111 — оба эти порта непосредственно связаны через сегменты *A* и *B* с коммутатором 111, а значит, получают пакеты Hello с метрикой, равной 0. Так как идентификатор порта 1 меньше идентификатора порта 2, то корневым портом коммутатора 222 выбирается порт 1.

   По аналогичной причине корневым портом коммутатора 555 становится порт 1, а не порт 2. Оба эти порта получают сообщения Hello, генерируемые корневым коммутатором 111, с наименьшим значением метрики 200 000. Порт 1 получает такие сообщения по маршруту: порт 1 коммутатора 111 — сегмент *C* — порт 1 коммутатора 333 — порт 6 коммутатора 333 — сегмент *G*; соответственно порт 2 получает их по маршруту: порт 3 коммутатора 111 — сегмент *E* — порт 1 коммутатора 444 — порт 6 коммутатора 444 — сегмент *I*.

3. *Выбор назначенных коммутаторов и портов для каждого сегмента сети.*

   Назначенным является тот коммутатор (из числа коммутаторов, непосредственно подключенных к данному сегменту), у которого расстояние до корневого моста является минимальным (точнее, расстояние от корневого порта этого коммутатора до корневого

коммутатора). Назначенные порты для сегментов исполняют ту же роль, что корневые порты для коммутаторов, — они находятся на кратчайшем пути до корневого коммутатора.

Как и при выборе корневого порта, здесь используется распределенная процедура. Каждый коммутатор сегмента прежде всего исключает из рассмотрения свой корневой порт (для сегмента, к которому он подключен, всегда существует другой коммутатор, расположенный ближе к корню). Для каждого из оставшихся портов выполняется сравнение принятых по ним минимальных расстояний до корня (еще до наращивания на метрику сегмента) с расстоянием до корня корневого порта данного коммутатора. Если все принятые на этом порту расстояния оказываются больше, чем расстояние от собственного корневого порта, значит, для сегмента, к которому подключен порт, кратчайший путь к корневому коммутатору проходит через него и он становится назначенным. Коммутатор делает все свои порты, для которых такое условие выполняется, назначенными. Когда имеется несколько портов с одинаковым кратчайшим расстоянием до корневого коммутатора, выбирается порт с наименьшим идентификатором.

В рассматриваемом примере коммутатор 111 при проверке порта 1 обнаруживает, что через этот порт принимаются пакеты с минимальным расстоянием 200 000 (это пакеты от порта 1 коммутатора 222, который ретранслирует через все свои порты сообщения Hello, полученные от коммутатора 111, но с измененной метрикой, в частности передает их и коммутатору 111). Так как коммутатор 111 является корневым, то его расстояние до корневого коммутатора равно нулю, то есть меньше, чем у получаемых через порт 1 сообщений. Поэтому коммутатор 1 объявляет свой порт 1 назначенным для сегмента *A*. Коммутатор 222 не может объявить свой порт 1 назначенным для сегмента *A*, так как через него он получает сообщения с минимальной метрикой 0, а у его корневого порта метрика равна 200 000.

На выполнение всех трех этапов коммутаторам сети отводится по умолчанию 15 секунд. Эта стадия работы портов называется стадией прослушивания (listening), поскольку порты слушают только сообщения BPDU и не передают пользовательских кадров. Считается, что порты находятся в заблокированном состоянии, которое относится только к пользовательским кадрам, в то время как кадры BPDU обрабатываются. Предполагается, что в стадии прослушивания каждый коммутатор получает столько пакетов Hello, сколько требуется для определения состояния своих портов.

Все остальные порты, кроме корневых и назначенных, каждым коммутатором блокируются и не могут передавать пользовательские кадры. Математически доказано, что при таком выборе активных портов из сети исключаются петли, а оставшиеся связи образуют *покрывающее дерево* (если оно вообще может быть построено при существующих связях в сети).

Результат работы протокола STP для нашего примера показан на рис. 13.3.

На рисунке корневые порты коммутаторов отмечены символом *R*, назначенные порты закрашены, а заблокированные зачеркнуты.

После построения покрывающего дерева коммутатор начинает принимать (но не продвигать) пакеты данных и на основе их адресов источника строить таблицу продвижения. Это обычный режим обучения прозрачного моста, который ранее нельзя было активизировать, так как порт не был уверен в том, что он останется корневым или назначенным и будет передавать пакеты данных.

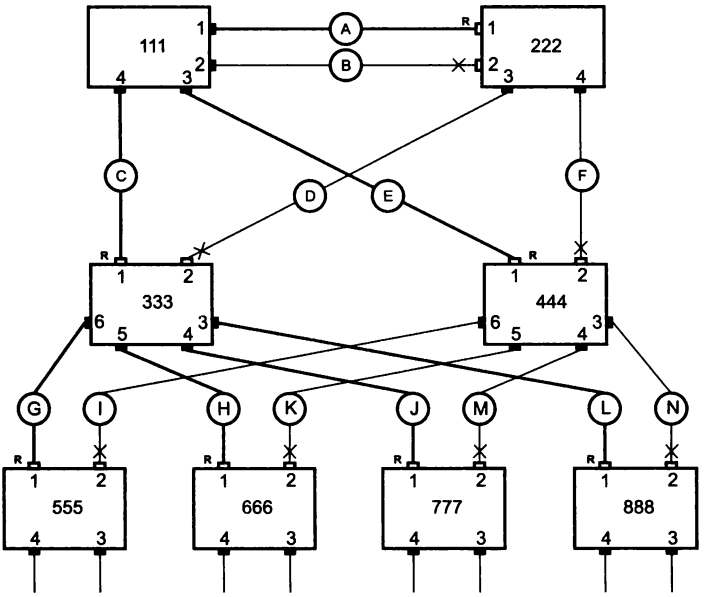


Рис. 13.3. Активная топология, найденная по протоколу STP

В процессе нормальной работы корневой коммутатор продолжает генерировать пакеты Hello, а остальные коммутаторы получают их через свои корневые порты и ретранслируют через назначенные порты. Если по истечении максимального времени жизни сообщения (по умолчанию — 10 интервалов Hello, то есть 20 с) корневой порт любого коммутатора сети не получает служебный пакет Hello, он инициализирует новую процедуру построения покрывающего дерева — тем самым обеспечивается отказоустойчивость локальной сети на коммутаторах. При этом на все порты генерируется и передается пакет Hello, в котором коммутатор указывает себя в качестве корневого. Аналогичным образом ведут себя и другие коммутаторы сети, у которых сработал таймер истечения максимального времени жизни сообщения, в результате чего выбирается новая активная конфигурация.

Версия RSTP

Основным недостатком протокола STP является его «медлительность»: в сетях с большим количеством коммутаторов время определения новой активной конфигурации может оказаться слишком большим. Если в сети используются заданные по умолчанию значения тайм-аутов, переход на новую конфигурацию может занять свыше 50 секунд: 20 секунд понадобится на констатацию факта потери связи с корневым коммутатором (истечение таймера — единственный способ узнать об этом событии в стандартном варианте STA), еще 2 × 15 секунд нужно для перехода портов в состояние продвижения.

Имеющиеся многочисленные нестандартные версии STA позволяют сократить время реконфигурирования за счет усложнения алгоритма, например добавления новых типов служебных сообщений. В 2001 году была разработана стандартная ускоренная версия протокола — RSTP (спецификация IEEE 802.1w), которая затем вошла в качестве раздела 17 в общий стандарт 802.1D-2004.

В версии RSTP для сокращения времени построения активной топологии использовано несколько новых механизмов и приемов.

Коммутаторы стали учитывать тип сегмента, подключенного к порту. Различаются следующие типы сегментов:

- *Двухточечный сегмент.* В коммутируемых сетях это единственный тип сегмента; для него у порта существует единственный порт-сосед.
- *Разделяемая среда.* Стандарт RSTP по-прежнему учитывает существование разделяемой среды, так как формально ее никто не отменял для скоростей ниже 10 Гбит/с.
- *Тупиковая связь* (edge port). Связь, которая соединяет порт коммутатора с конечным узлом сети; по этому сегменту нет смысла ожидать прихода сообщений протокола RSTP. Тупиковая связь конфигурируется администратором.

В случае подключения к порту тупикового сегмента этот порт не участвует в работе протокола RSTP, а сразу после включения переходит в стадию продвижения кадров. Нужно заметить, что в стандарте RSTP начальное заблокированное состояние портов переименовано в состояние отбрасывания.

Для портов со связями остальных типов переход в состояние продвижения по-прежнему достижим только после прохождения стадии обучения.

Исключается стадия прослушивания. Коммутаторы не делают паузу в 15 секунд для того, чтобы зафиксировать соответствующую роль порта, например корневого или назначенного. Вместо этого порты переходят в стадию обучения сразу же после назначения им роли корневого или назначенного порта.

Сокращается период фиксации отказа в сети — вместо 10 периодов неполучения сообщений Hello он стал равен трем таким периодам, то есть 6 секунд вместо 20.

Введены новые роли портов — появились **альтернативный** (alternative) и **резервный** (backup) порты. Альтернативный порт является портом-дублером корневого порта коммутатора, то есть он начинает продвигать кадры в том случае, когда отказывает (либо перестает принимать сообщения Hello в течение трех периодов) корневой порт. Резервный порт является портом-дублером назначенного порта сегмента; однако такая роль порта имеет смысл только для сегментов, представляющих собой разделяемую среду. Альтернативные и резервные порты находятся в состоянии отбрасывания кадров, так как они не должны продвигать кадры до тех пор, пока их роль не изменится на роль корневого или назначенного порта.

Как альтернативные, так и резервные порты выбираются одновременно с корневыми и назначенными портами. Такой подход значительно ускоряет реакцию сети на отказы, так как переход, например, на альтернативный порт происходит сразу же после фиксации отказа и не связан с ожиданием истечения тайм-аутов. За счет новых механизмов и новых ролей портов протокол RSTP строит новую активную топологию существенно быстрее, чем протокол STP, — за несколько секунд вместо минуты или даже нескольких минут. Протокол RSTP совместим с протоколом STP, так что сеть, построенная из коммутаторов, часть из которых поддерживает RSTP, а часть — STP, будет работать нормально.

Фильтрация трафика

Локальная сеть обеспечивает взаимодействие каждого узла с каждым — это очень полезное свойство, так как не требуется производить никаких специальных действий, чтобы обес-

печить доступ узла *A* к узлу *B*, — достаточно того, что эти узлы подключены к одной и той же локальной сети. В то же время в сети могут возникать ситуации, когда такая тотальная доступность узлов нежелательна. Примером может служить сервер финансового отдела, доступ к которому желательно разрешить только с компьютеров нескольких конкретных сотрудников этого отдела. Конечно, доступ можно ограничить на уровне операционной системы или системы управления базой данных самого сервера, но для надежности желательно иметь несколько эшелонов защиты и ограничить доступ еще и на уровне сетевого трафика.

Многие модели коммутаторов позволяют администраторам задавать дополнительные условия фильтрации кадров наряду со стандартными условиями их фильтрации в соответствии с информацией адресной таблицы. Такие фильтры называют пользовательскими.

Пользовательский фильтр, который также часто называют **списком доступа** (access list), предназначен для создания дополнительных барьеров на пути кадров, что позволяет ограничивать доступ определенных групп пользователей к отдельным службам сети. Пользовательский фильтр — это набор условий, которые ограничивают обычную логику передачи кадров коммутаторами.

Наиболее простыми являются пользовательские фильтры на основе MAC-адресов станций. Так как MAC-адреса — это та информация, с которой работает коммутатор, он позволяет создавать подобные фильтры удобным для администратора способом, возможно, проставляя некоторые условия в дополнительном поле адресной таблицы, например условие отбрасывать кадры с определенным адресом (см. рис. 12.5 в главе 12). Таким способом пользователю, работающему на компьютере с данным MAC-адресом, полностью запрещается доступ к ресурсам другого сегмента сети.

Рассмотрим применение пользовательского фильтра на примере сети, показанной на рис. 13.4.

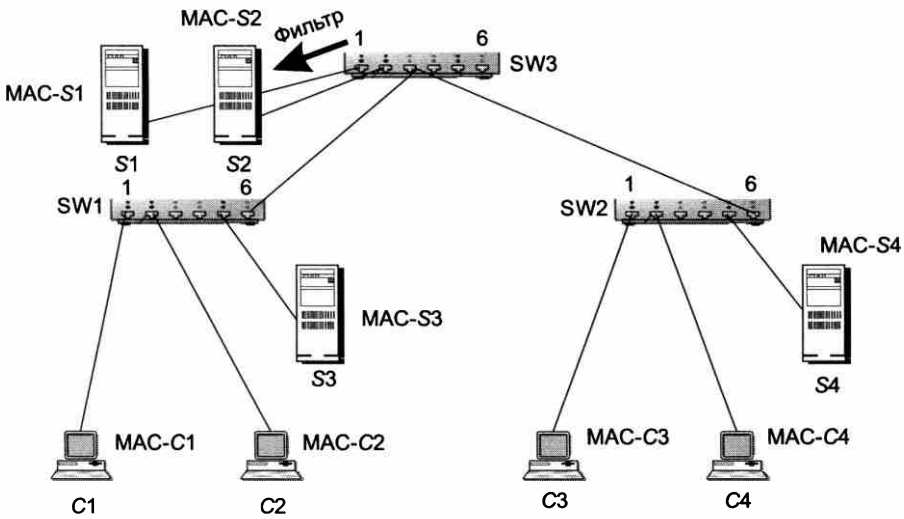


Рис. 13.4. Контроль доступа к серверу с помощью пользовательского фильтра

Пусть мы хотим разрешить доступ к серверу *S*1 только с компьютеров *C*1 и *C*3, кадры от всех остальных компьютеров до этого сервера доходить не должны. Список доступа, который решает эту задачу, может выглядеть так:

```
10 permit MAC-C1 MAC-S1
20 permit MAC-C3 MAC-S1
30 deny any any
```

Числа 10, 20 и 30 — это номера строк данного списка. Строки нумеруются с интервалом 10, чтобы в дальнейшем была возможность добавить в этот список другие записи, сохраняя исходную последовательность строк. Первое условие разрешает (`permit`) передачу кадра, если его адрес источника равен MAC-C1, а адрес назначения — MAC-S1; второе условие делает то же, но для кадра с адресом источника MAC-C3, третье условие запрещает (`deny`) передачу кадров с любыми (`any`) адресами.

Для того чтобы список доступа начал работать, его нужно применить к трафику определенного направления на какому-либо порту коммутатора: либо к входящему, либо к исходящему. В нашем примере нам нужно применить список доступа к исходящему трафику порта 1 коммутатора SW3, к которому подключен сервер *S*1. Коммутатор SW3 перед тем, как предать кадр на порт 1, будет просматривать условия списка доступа по очереди. Если какое-то условие из списка соблюдается, то коммутатор выполняет действие этого условия для обрабатываемого кадра, и на этом применение списка доступа к данному кадру заканчивается.

Поэтому когда от компьютера *C*1 приходит кадр, адресованный серверу *S*1, то соблюдается первое условие списка, которое разрешает передачу кадра, так что коммутатор выполняет стандартное действие по продвижению кадра и тот доходит до сервера *S*2. С кадром от компьютера *C*3 совпадение происходит при проверке второго условия, и он также передается. Однако когда приходят кадры от других компьютеров, например компьютера *C*2, то ни первое, ни второе условие не соблюдается, зато соблюдается третье условие, поэтому кадр не передается, а отбрасывается.

Списки доступа коммутаторов не работают с широковещательными адресами Ethernet, такие кадры всегда передаются на все порты коммутатора. Списки доступа коммутаторов достаточно примитивны, поскольку способны оперировать только информацией канального уровня, то есть MAC-адресами. Списки доступа маршрутизаторов гораздо более гибкие и мощные, поэтому на практике они применяются гораздо чаще[1].

Иногда администратору требуется задать более тонкие условия фильтрации, например запретить некоторому пользователю печатать свои документы на Windows-сервере печати, находящемуся в чужом сегменте, а остальные ресурсы этого сегмента сделать доступными. Для реализации подобного фильтра нужно запретить передачу кадров, которые удовлетворяют следующим условиям: во-первых, имеют определенный MAC-адрес, во-вторых, содержат в поле данных пакеты SMB, в-третьих, в соответствующем поле этих пакетов в качестве типа сервиса указана печать. Коммутаторы не анализируют протоколы верхних уровней, такие как SMB, поэтому администратору приходится для задания условий фильтрации «вручную» определять поле, по значению которого нужно осуществлять фильтрацию. В качестве признака фильтрации администратор указывает пару «смещение-

[1] Существуют так называемые коммутаторы 3-го уровня, которые объединяют функции коммутаторов и маршрутизаторов (в несколько усеченном виде) и позволяют фильтровать трафик с привлечением условий на уровне как MAC-адресов, так и IP-адресов.

размер» относительно начала поля данных кадра канального уровня, а затем приводит еще шестнадцатеричное значение этого поля.

Сложные условия фильтрации обычно записываются в виде булевых выражений, формируемых с помощью логических операторов AND и OR.

Виртуальные локальные сети

Важным свойством коммутатора локальной сети является способность контролировать передачу кадров между сегментами сети. По различным причинам (соблюдение прав доступа, политика безопасности и т. д.) некоторые кадры не следует передавать по адресу назначения.

Как мы выяснили в предыдущем разделе, ограничения такого типа можно реализовать с помощью *пользовательских фильтров*. Однако пользовательский фильтр может запретить коммутатору передачу кадров только по конкретным адресам, а широковещательный трафик он *обязан* передать всем сегментам сети. Так требует алгоритм его работы. Поэтому, как уже отмечалось, сети, созданные на основе коммутаторов, иногда называют *плоскими* – из-за отсутствия барьеров на пути широковещательного трафика. Технология виртуальных локальных сетей позволяет преодолеть указанное ограничение.

Виртуальной локальной сетью (Virtual Local Area Network, VLAN) называется группа узлов сети, трафик которой, в том числе широковещательный, на канальном уровне полностью изолирован от трафика других узлов сети.

Это означает, что передача кадров между разными виртуальными сетями на основании адреса канального уровня невозможна независимо от типа адреса (уникального, группового или широковещательного). В то же время внутри виртуальной сети кадры передаются по технологии коммутации, то есть только на тот порт, который связан с адресом назначения кадра.

Виртуальные локальные сети могут *перекрываться*, если один или несколько компьютеров входят в состав более чем одной виртуальной сети. На рис. 13.5 сервер электронной почты входит в состав виртуальных сетей 3 и 4. Это означает, что его кадры передаются коммутаторами всем компьютерам, входящим в эти сети. Если же какой-то компьютер входит в состав только виртуальной сети 3, то его кадры до сети 4 доходить не будут, но он может взаимодействовать с компьютерами сети 4 через общий почтовый сервер. Такая схема защищает виртуальные сети друг от друга не полностью, например широковещательный шторм, возникший на сервере электронной почты, затопит и сеть 3, и сеть 4.

Говорят, что виртуальная сеть образует *домен широковещательного трафика* по аналогии с доменом коллизий, который образуется повторителями сетей Ethernet.

Назначение виртуальных сетей

Как мы видели на примере из предыдущего раздела, с помощью пользовательских фильтров можно вмешиваться в нормальную работу коммутаторов и ограничивать взаимодействие узлов локальной сети в соответствии с требуемыми правилами доступа. Однако механизм пользовательских фильтров коммутаторов имеет несколько недостатков:

- *Приходится задавать отдельные условия для каждого узла сети*, используя при этом громоздкие MAC-адреса. Гораздо проще было бы группировать узлы и описывать условия взаимодействия сразу для групп.
- *Невозможно блокировать широковещательный трафик*. Широковещательный трафик может быть причиной недоступности сети, если какой-то ее узел умышленно или неумышленно с большой интенсивностью генерирует широковещательные кадры.

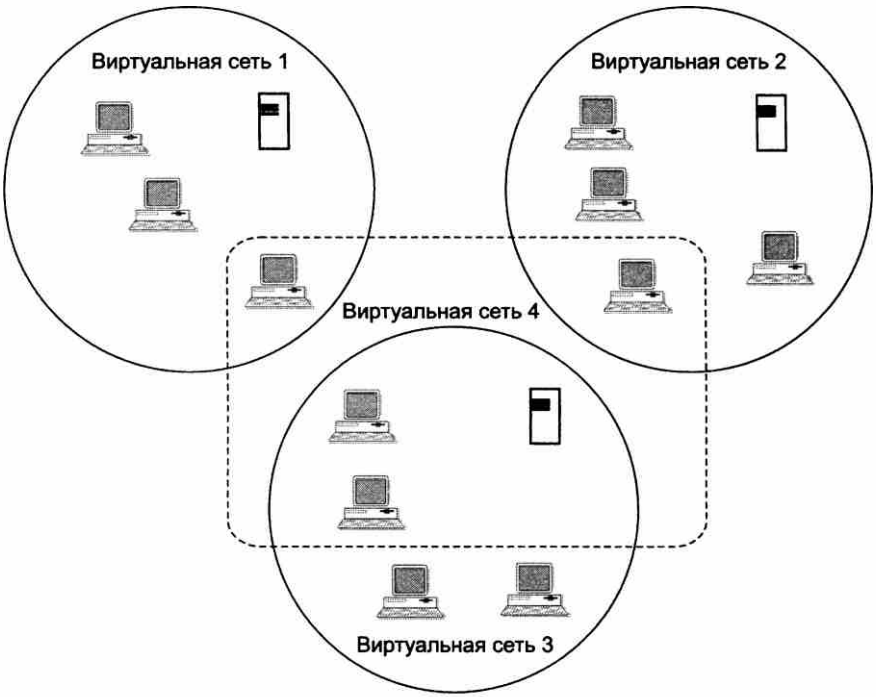


Рис. 13.5. Виртуальные локальные сети

Техника виртуальных локальных сетей решает задачу ограничения взаимодействия узлов сети другим способом.

Основное назначение технологии VLAN состоит в облегчении процесса создания изолированных сетей, которые затем обычно связываются между собой с помощью маршрутизаторов. Такое построение сети создает мощные барьеры на пути нежелательного трафика из одной сети в другую. Сегодня считается очевидным, что любая крупная сеть должна включать маршрутизаторы, иначе потоки ошибочных кадров, например широковещательных, будут периодически «затапливать» всю сеть через прозрачные для них коммутаторы, приводя ее в неработоспособное состояние.

Достоинством технологии виртуальных сетей является то, что она позволяет создавать полностью изолированные сегменты сети путем логического конфигурирования коммутаторов, не прибегая к изменению физической структуры.

До появления технологии VLAN для создания отдельной сети использовались либо физически изолированные сегменты коаксиального кабеля, либо не связанные между собой сегменты, построенные на повторителях и мостах. Затем эти сети связывались маршрутизаторами в единую составную сеть (рис. 13.6).

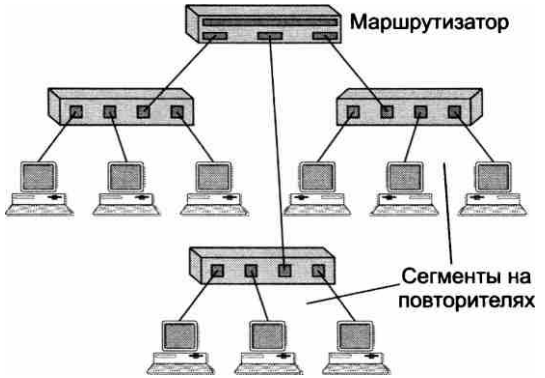


Рис. 13.6. Составная сеть, состоящая из сетей, построенных на основе повторителей

Изменение состава сегментов (переход пользователя в другую сеть, дробление крупных сегментов) при таком подходе подразумевает физическую перекоммутацию разъемов на передних панелях повторителей или на кроссовых панелях, что не очень удобно в больших сетях — много физической работы, к тому же высока вероятность ошибки.

Для связывания виртуальных сетей в общую сеть требуется привлечение средств сетевого уровня. Он может быть реализован в отдельном маршрутизаторе или в составе программного обеспечения коммутатора, который тогда становится комбинированным устройством — так называемым **коммутатором 3-го уровня**.

Технология виртуальных сетей долгое время не стандартизировалась, хотя и была реализована в очень широком спектре моделей коммутаторов разных производителей. Положение изменилось после принятия в 1998 году стандарта IEEE 802.1Q, который определяет базовые правила построения виртуальных локальных сетей, не зависящие от протокола канального уровня, поддерживаемого коммутатором.

Создание виртуальных сетей на базе одного коммутатора

При создании виртуальных сетей на основе одного коммутатора обычно используется механизм *группирования портов* коммутатора (рис. 13.7). При этом каждый порт приписывается той или иной виртуальной сети. Кадр, пришедший от порта, принадлежащего, например, виртуальной сети 1, никогда не будет передан порту, который не принадлежит этой виртуальной сети. Порт можно приписать нескольким виртуальным сетям, хотя на практике так делают редко — пропадает эффект полной изоляции сетей.

Создание виртуальных сетей путем группирования портов не требует от администратора большого объема ручной работы — достаточно каждый порт приписать к одной из

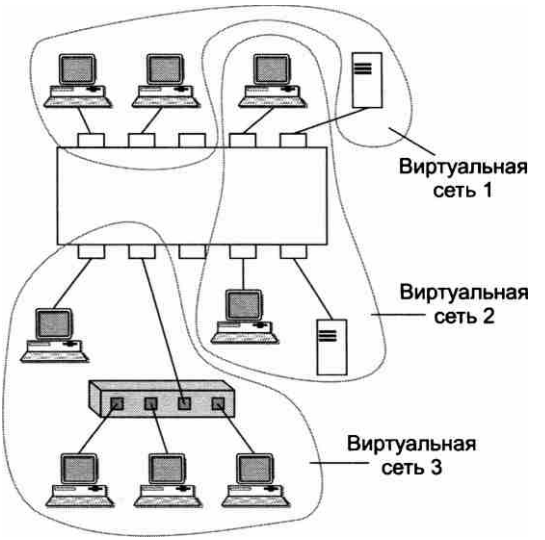


Рис. 13.7. Виртуальные сети, построенные на одном коммутаторе

нескольких заранее поименованных виртуальных сетей. Обычно такая операция выполняется с помощью специальной программы, прилагаемой к коммутатору.

Второй способ образования виртуальных сетей основан на *группировании MAC-адресов*. Каждый MAC-адрес, который изучен коммутатором, приписывается той или иной виртуальной сети. При существовании в сети множества узлов этот способ требует от администратора большого объема ручной работы и по этой причине не получил распространения.

Создание виртуальных сетей на базе нескольких коммутаторов

Рисунок 13.8 иллюстрирует проблему, возникающую при создании виртуальных сетей на основе нескольких коммутаторов, поддерживающих технику *группирования портов*.

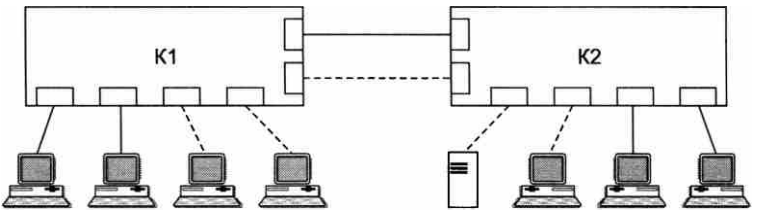


Рис. 13.8. Построение виртуальных сетей на нескольких коммутаторах с группированием портов

Если узлы какой-либо виртуальной сети подключены к разным коммутаторам, то для подключения каждой такой сети на коммутаторах должна быть выделена специальная пара портов. В противном случае, если коммутаторы будут связаны только одной парой портов,

информация о принадлежности кадра той или иной виртуальной сети при передаче из коммутатора в коммутатор будет утеряна. Таким образом, коммутаторы с группированием портов требуют для своего соединения столько портов, сколько виртуальных сетей они поддерживают. Порты и кабели используются в этом случае очень расточительно. Кроме того, при соединении виртуальных сетей через маршрутизатор для каждой виртуальной сети выделяются отдельные кабель и порт маршрутизатора, что также приводит к большим накладным расходам.

Группирование MAC-адресов в виртуальную сеть на каждом коммутаторе избавляет от необходимости связывать их по нескольким портам, поскольку в этом случае MAC-адрес становится меткой виртуальной сети. Однако этот способ требует выполнения большого количества ручных операций по маркировке MAC-адресов на каждом коммутаторе сети.

Описанные два подхода основаны только на добавлении дополнительной информации к адресным таблицам коммутатора, и в них отсутствует возможность встраивания в передаваемый кадр информации о принадлежности кадра виртуальной сети.

Поэтому широкое распространение получил иной подход, основанный на введении в кадр *дополнительного поля*, которое хранит информацию о принадлежности кадра той или иной виртуальной локальной сети при его перемещениях между коммутаторами сети. При этом нет необходимости помнить в каждом коммутаторе о принадлежности всех MAC-адресов составной сети виртуальным сетям.

Дополнительное поле с пометкой о номере виртуальной сети используется только тогда, когда кадр передается от коммутатора к коммутатору, а при передаче кадра конечному узлу оно обычно удаляется. При этом модифицируется протокол взаимодействия «коммутатор—коммутатор», а программное и аппаратное обеспечение конечных узлов остается неизменным.

До принятия стандарта IEEE 802.1Q существовало много фирменных протоколов этого типа, но все они имели один недостаток — оборудование различных производителей при образовании VLAN оказывалось несовместимым.

Стандарт IEEE 802.1Q вводит в кадре Ethernet дополнительный заголовок, который называется тегом виртуальной локальной сети.

Тег виртуальной локальной сети состоит из поля **TCI** (Tag Control Information — управляющая информация тега) размером в 2 байта и предшествующего ему поля **EtherType**, которое является стандартным для кадров Ethernet и также состоит из 2 байт (рис. 13.9).

Тег VLAN не является обязательным для кадров Ethernet. Кадр, у которого имеется такой заголовок, называют **помеченным** (tagged frame). Коммутаторы могут одновременно работать как с помеченными, так и с непомеченными кадрами. Из-за добавления тега VLAN максимальная длина поля данных уменьшилась на 4 байта.

Для того чтобы оборудование локальных сетей могло отличать и понимать помеченные кадры, для них введено специальное значение поля EtherType, равное 0x8100. Это значение говорит о том, что за ним следует поле TCI, а не стандартное поле данных. Обратите внимание, что в помеченном кадре за полями тега VLAN следует другое поле EtherType, указывающее тип протокола, данные которого переносятся полем данных кадра.

В поле TCI находится 12-битное поле номера (идентификатора) VLAN, называемого *VID*. Разрядность поля VID позволяет коммутаторам создавать до 4096 виртуальных сетей.

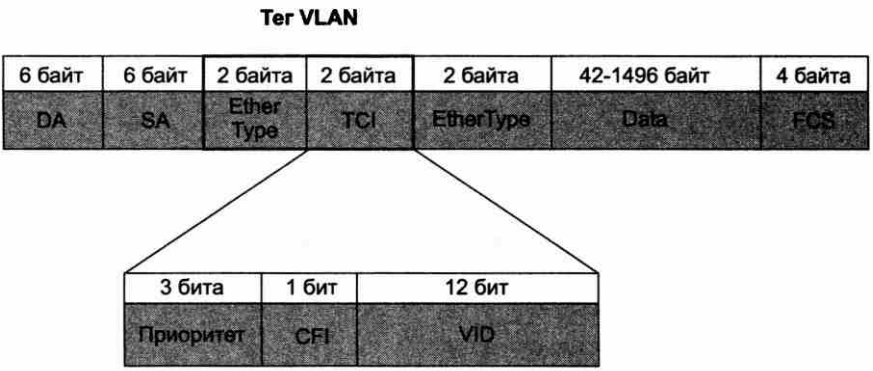


Рис. 13.9. Структура помеченного кадра Ethernet

Помимо этого, в поле TCI помещено 3-битное поле *приоритета* кадра. Однобитное поле *CFI* было введено с целью поддержания специального формата кадра Token Ring, для сетей Ethernet оно должно содержать значение 0.

Пользуясь значением VID в помеченных кадрах, коммутаторы сети выполняют групповую фильтрацию трафика, разбивая сеть на виртуальные сегменты, то есть на VLAN. Для поддержки этого режима каждый порт коммутатора приписывается к одной или нескольким виртуальным локальным сетям, то есть выполняется группировка портов.

Поле приоритета предназначено для согласованного обеспечения качества обслуживания (QoS) различных классов трафика. Всего может поддерживаться до восьми классов трафика (это определяется тремя битами поля), при этом коммутаторы могут применять все методы обеспечения QoS, описанные в главе 6.

Конфигурирование VLAN

Существуют различные подходы к конфигурированию виртуальных локальных сетей, построенных на нескольких коммутаторах.

Схема «транк — линия доступа»

Наиболее распространенным является подход, основанный на понятиях линии доступа и транка.

Линия доступа связывает порт коммутатора (называемый в этом случае **портом доступа**) с конечным узлом (компьютером, мобильным устройством и т. п.), принадлежащим некоторой виртуальной локальной сети. Предполагается, что конечный узел работает с непомеченными кадрами, то есть структура VLAN для него прозрачна.

Транк — это линия связи, которая соединяет между собой порты двух коммутаторов; в общем случае через транк передается трафик нескольких виртуальных сетей.

Коммутаторы, поддерживающие технику VLAN, без специального конфигурирования по умолчанию работают как стандартные коммутаторы, обеспечивая соединения всех со всеми. В сети, образованной такими коммутаторами, все конечные узлы по умолчанию относятся к условной сети VLAN1 с идентификатором VID, равным 1. Все порты этой сети, к которым подключены конечные узлы, по определению являются портами доступа. Сеть VLAN1 можно отнести к виртуальным локальным сетям лишь *условно*, так как по ней передаются непомеченные кадры. Условная сеть VLAN также называется сетью VLAN, предлагаемой по умолчанию (default VLAN) или естественной (native VLAN).

Для того чтобы образовать в исходной сети виртуальную локальную сеть, нужно в первую очередь выбрать для нее значение идентификатора VID, отличное от 1, а затем, используя команды конфигурирования коммутатора, приписать к этой сети те порты, к которым присоединены включаемые в нее компьютеры. Порт доступа может быть приписан только к одной виртуальной локальной сети.

Порты доступа получают от конечных узлов сети непомеченные кадры и помечают их тегом VLAN, содержащим то значение VID, которое назначено этому порту. При передаче же помеченных кадров конечному узлу порт доступа удаляет тег виртуальной локальной сети.

Для более наглядного описания вернемся к рассмотренному ранее примеру сети. На рис. 13.10 показано, как решается задача избирательного доступа к серверам на основе техники VLAN.

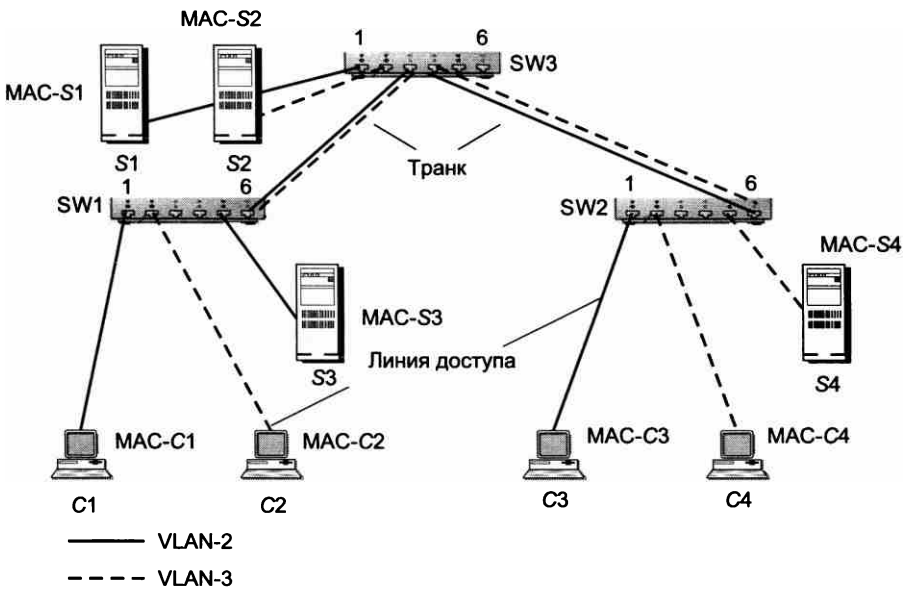


Рис. 13.10. Разбиение сети на две виртуальные локальные сети

Будем считать, что поставлена задача обеспечить доступ компьютеров *C*1 и *C*3 к серверам *S*1 и *S*3, в то время как компьютеры *C*2 и *C*4 должны иметь доступ только к серверам *S*2 и *S*4.

Чтобы решить эту задачу, можно организовать две виртуальные локальные сети, VLAN2 и VLAN3 (напомним, что сеть VLAN1 уже существует по умолчанию — это наша исходная сеть), приписав один набор компьютеров и серверов к VLAN2, а другой — к VLAN3. Первым шагом в конфигурировании VLAN2 и VLAN3 является их активизация в каждом из коммутаторов сети.

Затем к этим сетям VLAN можно приписать определенные порты, работающие в режиме доступа. Для приписывания конечных узлов к определенной виртуальной локальной сети соответствующие порты объявляются портами доступа этой сети путем назначения им соответствующего идентификатора VID. Например, порт 1 коммутатора SW1 должен быть объявлен портом доступа VLAN2 путем назначения ему идентификатора VID2, то же самое должно быть проделано с портом 5 коммутатора SW1, портом 1 коммутатора SW2 и портом 1 коммутатора SW3. Порты доступа сети VLAN3 должны получить идентификатор VID3.

В нашей сети нужно также организовать транки — те линии связи, которые соединяют между собой порты коммутаторов. Порты, подключенные к транкам, не добавляют и не удаляют теги, они просто передают кадры в неизменном виде. В нашем примере такими портами должны быть порты 6 коммутаторов SW1 и SW2, а также порты 3 и 4 коммутатора SW3. Порты в нашем примере должны поддерживать сети VLAN2 и VLAN3 (и VLAN1, если в сети есть узлы, явно не приписанные ни к одной виртуальной локальной сети). Транк может быть сконфигурирован как в «неразборчивом» режиме, когда он передает кадры с любым номером VLAN, так и в избирательном режиме, когда он передает кадры только определенных номеров VLAN.

Коммутаторы, поддерживающие технологию VLAN, осуществляют дополнительную фильтрацию трафика. В том случае если таблица продвижения коммутатора говорит о том, что пришедший кадр нужно передать на некоторый порт, перед передачей коммутатор проверяет, соответствует ли значение VID в теге VLAN кадра той виртуальной локальной сети, которая приписана к этому порту. В случае соответствия кадр передается, несоответствия — отбрасывается. Непомеченные кадры обрабатываются аналогичным образом, но с использованием условной сети VLAN1. MAC-адреса изучаются коммутаторами сети отдельно по каждой виртуальной локальной сети.

Схема с гибким конфигурированием портов

В этой схеме порты не делятся на транки и порты доступа, каждый порт может быть гибко сконфигурирован для специфической поддержки кадров VLAN в зависимости от потребностей сети. Порт может работать в следующих режимах:

- *Принимать только непомеченные кадры.* В этом случае режим соответствует режиму порта доступа.
- *Принимать только помеченные кадры.* При этом порту могут быть приписаны один или несколько номеров VLAN. Этот режим соответствует избирательному режиму работы транка. Помеченные кадры передаются без отбрасывания/добавления тега VLAN.
- *Принимать как помеченные, так и непомеченные кадры.* Непомеченные кадры всегда принадлежат естественной сети VLAN 1 (некоторые модели коммутаторов позволяют администратору назначить естественной сети VLAN произвольный номер, отличный от 1). Порту может быть приписан один или несколько номеров VLAN.

В том случае, когда коммутатор поддерживает образование нескольких логических портов для одного и того же физического порта, каждый логический порт может работать в собственном режиме.

В схеме с гибким конфигурированием портов администратору проще производить изменения в конфигурации виртуальных локальных сетей, так как ему не требуется изменять роли портов в сети (например, изменять роль порта доступа на роль транка), но при этом увеличивается объем конфигурационных операций.

Автоматизации конфигурирования VLAN

В сети, состоящей из большого количества коммутаторов и не разделенной на подсети маршрутизаторами, полностью ручное конфигурирование VLAN может приводить к ошибкам из-за несогласованности информации об активных сетях VLAN на различных коммутаторах, особенно если их конфигурируют разные администраторы.

Существует несколько протоколов, позволяющих частично автоматизировать конфигурирование VLAN в сети.

Cisco VLAN Trunking Protocol. Этот протокол является фирменным протоколом компании Cisco и работает только на коммутаторах этой компании. Коммутаторы Cisco поддерживают модель «транк — линии доступа», а протокол VTP позволяет по транковым связям передавать информацию о сетях VLAN, активизированных на одном из коммутаторов, другим коммутаторам сети. Поэтому администратору достаточно добавить (или удалить) VLAN на одном из коммутаторов сети, после чего все остальные коммутаторы сети получат информацию о добавлении (удалении) VLAN с данным номером и произведут соответствующие изменения в своих конфигурационных записях.

Для удобства администрирования больших сетей в протоколе VTP существует понятие домена — все сообщения протокола VTP воспринимаются коммутаторами только одного и того же домена (имя домена и его пароль конфигурируются на каждом коммутаторе вручную). Приписывание VLAN порту доступа при работе протокола VTP по-прежнему выполняется вручную. Порты, работающие в режиме транка, приписывают номера VLAN к транку (работающему в избирательном режиме) динамически. При этом протокол VTP автоматически выполняет отсечение номера VLAN для транков некоторого домена, если в данном домене этот номер не приписан ни одному из его портов доступа (это свойство называется VTP pruning).

Свои VTP-объявления коммутаторы рассылают с использованием группового адреса.

GARP VLAN Registration Protocol (GVRP). Этот протокол является одним из двух популярных приложений протокола GARP (Generic Attribute Registration Protocol). Протокол GARP был разработан рабочей группой IEEE 802.1 для того, чтобы коммутаторы локальной сети могли сообщать друг другу (регистрировать в сети) различные атрибуты. На практике этот протокол стали применять для регистрации двух типов атрибутов: достижимых через некоторый порт коммутатора групповых MAC-адресов и номеров VLAN. Соответственно появились два приложения протокола GARP: GMRP (GARP Multicast Registration Protocol) и GVRP (GARP VLAN Registration Protocol). GMRP позволяет коммутаторам отсечь бесполезный трафик с групповыми адресами от сегментов сети, в которых нет активных получателей этих адресов.

Назначение GVRP примерно то же, что у протокола Cisco VTP, — он позволяет конфигурировать новую сеть VLAN только на одном из коммутаторов большой сети, остальные коммутаторы выполняют изменения в своей конфигурации автоматически, получая сообщения GVRP. GVRP является стандартным протоколом и поэтому работает на коммутаторах различных производителей в отличие от фирменного протокола Cisco VTP. Кроме того, в отличие от VTP он может работать не только на портах-транках, но и на портах доступа, к тому же конечные узлы также могут поддерживать GVRP, а значит, сетевой адаптер компьютера может инициировать динамическое приписывание номера своей сети VLAN у порта доступа.

Multiple VLAN Registration Protocol (*MVRP*). Протокол GARP обладал несколькими существенными недостатками — в больших сетях он порождал большое количество служебного трафика, кроме того, процесс установления новой конфигурации мог длиться слишком долго из-за нескольких обязательных тайм-аутов. Поэтому в 2007 году группа IEEE 802.1 заменила GARP протоколом MRP (Multiple Registration Protocol). Соответственно протокол MVRP заменил GVRP (а MMRP — GMRP). За счет изменения формата сообщений и логики обмена ими служебный трафик был сокращен, а время установления новой конфигурации — уменьшено.

Альтернативные маршруты в виртуальных локальных сетях

По умолчанию протокол STP/RSTP образует в сети одно покрывающее дерево для всех виртуальных локальных сетей. Чтобы в сети можно было использовать разные покрывающие деревья для разных виртуальных локальных сетей, существует специальная версия протокола, называемая **множественным протоколом покрывающего дерева** (Multiple Spanning Tree Protocol, MSTP).

Протокол MSTP позволяет создать несколько покрывающих деревьев и приписывать к ним различные виртуальные локальные сети. Обычно создается небольшое количество деревьев, например два или три, чтобы сбалансировать нагрузку на коммутаторы, в противном случае, как мы видели в примере на рис. 13.2 и 13.3, единственное покрывающее дерево может полностью оставить без работы некоторые коммутаторы сети, то есть недоиспользовать имеющиеся сетевые ресурсы.

Если вернуться к нашему примеру (см. рис. 13.2), то при создании двух покрывающих деревьев можно сконфигурировать приоритеты коммутаторов так, чтобы для одного дерева корневым коммутатором стал коммутатор 111, а для второго — коммутатор 222 (рис. 13.11).

В этом варианте мы подразумеваем, что порты четырех коммутаторов с 555 по 888 сконфигурированы как порты доступа одной виртуальной локальной сети, например VLAN100, а порты трех тех же коммутаторов — как порты доступа другой виртуальной локальной сети, например VLAN200. Сеть VLAN100 приписана к покрывающему дереву с корневым коммутатором 111, а VLAN200 — к покрывающему дереву с корневым коммутатором 222. В этом варианте все коммутаторы сети используются для передачи трафика, что повышает производительность сети.

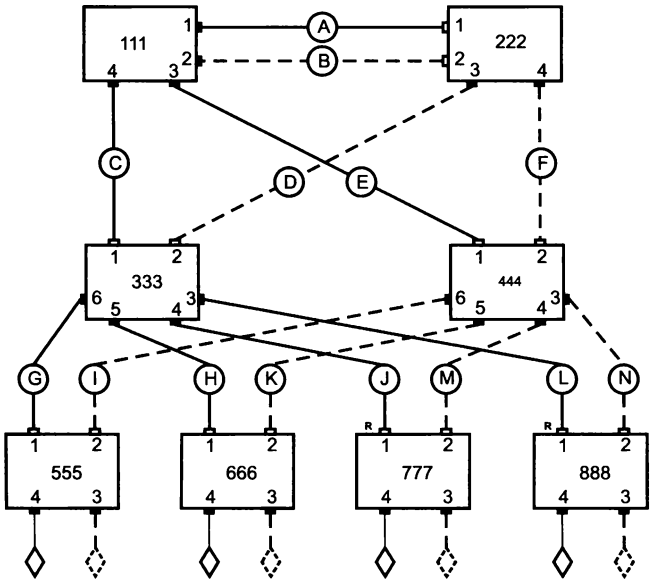


Рис. 13.11. Два покрывающих дерева, построенные по протоколу MSTP

Протокол MSTP основан на протоколе RSTP, поэтому обеспечивает быструю реакцию сети на отказы.

Ограничения коммутаторов

Применение коммутаторов позволяет преодолеть ограничения, свойственные сетям с разделяемой средой. Коммутируемые локальные сети могут покрывать значительные территории, плавно переходя в сети мегаполисов; они могут состоять из сегментов различной пропускной способности, образуя сети с очень высокой производительностью; они могут использовать альтернативные маршруты для повышения надежности и производительности. Однако построение сложных сетей без маршрутизаторов, а только на основе коммутаторов, имеет существенные ограничения.

- Серьезные ограничения по-прежнему накладываются на топологию коммутируемой локальной сети. Требование *отсутствия петель* преодолевается с помощью техники STP/RSTP/MSTP и агрегирования каналов лишь частично. Действительно, STP не позволяет задействовать все альтернативные маршруты для передачи пользовательского трафика, а агрегирование каналов разрешает так делать только на участках сети между двумя соседними коммутаторами. Подобные ограничения не позволяют применять многие эффективные топологии, пригодные для передачи трафика.
- Логические сегменты сети, расположенные между коммутаторами, *слабо изолированы* друг от друга, а именно — не защищены от так называемых широковещательных штормов. Использование же механизма виртуальных сетей, реализованного во многих коммутаторах, хотя и позволяет достаточно гибко создавать изолированные по трафику

группы станций, изолирует их полностью, то есть так, что узлы одной виртуальной сети не могут взаимодействовать с узлами другой виртуальной сети.

- В сетях, построенных на основе мостов и коммутаторов, достаточно *сложно решается задача фильтрации трафика* на основе данных, содержащихся в пакете. В таких сетях фильтрация выполняется только с помощью пользовательских фильтров, для создания которых администратору приходится иметь дело с двоичным представлением содержимого пакетов.
- Реализация транспортной подсистемы только средствами физического и канального уровней приводит к *недостаточно гибкой одноуровневой системе адресации*: в качестве адреса назначения используется MAC-адрес, жестко связанный с сетевым адаптером.
- У коммутаторов *ограничены возможности по трансляции протоколов* при создании гетерогенной сети. Они не могут транслировать протоколы WAN в протоколы LAN из-за различий в системе адресации этих сетей, а также различных значений максимального размера поля данных.

Наличие серьезных ограничений у протоколов канального уровня показывает, что построение на основе средств этого уровня больших неоднородных сетей является весьма проблематичным. Естественное решение в этих случаях — привлечение средств более высокого сетевого уровня.

Выводы

Для автоматического поддержания в сложных сетях резервных связей в коммутаторах реализуется алгоритм покрывающего дерева. Этот алгоритм описан в документе IEEE 802.1D и основан на периодическом обмене коммутаторов специальными кадрами, с помощью которых выявляются и блокируются петлевидные связи в сети.

Протокол STA находит конфигурацию покрывающего дерева за три этапа. На первом этапе определяется корневой коммутатор, на втором — корневые порты, на третьем — назначенные порты сегментов.

Недостатком протокола STA 802.1D является сравнительно большое время установления новой активной конфигурации — около 50 с. Новый стандарт RSTP устраняет этот недостаток за счет предварительного выбора портов-дублеров для корневых и назначенных портов, а также введения некоторых других новых механизмов.

Технология виртуальных локальных сетей (VLAN) позволяет в сети, построенной на коммутаторах, программным путем создать изолированные группы конечных узлов, между которыми отсутствует любой трафик, в том числе широковещательный.

Конфигурирование VLAN обычно ведется путем группирования портов или MAC-адресов.

Для построения виртуальной локальной сети на основе нескольких коммутаторов желательно помечать передаваемые кадры специальной меткой — тегом, идентифицирующим номер сети, которой принадлежит отправитель кадра. Стандартный формат тега VLAN определен в спецификации 802.1Q.

Протокол MSTP позволяет организовать в сети отдельные покрывающие деревья для виртуальных локальных сетей.

Контрольные вопросы

1. Каждый ли коммутатор, участвующий в построении покрывающего дерева, имеет корневой порт?
2. Может ли администратор влиять на выбор корневого коммутатора?
3. За счет каких усовершенствований протокол RSTP работает быстрее протокола STP? Варианты ответов:
   а) применение более быстрых процессоров коммутаторов;
   б) исключение тупиковых портов из процесса выбора корневых и назначенных портов;
   в) выбор портов-дублеров для корневых и назначенных портов.
4. Преимуществами разбиения локальной сети на VLAN являются:
   а) локализация широковещательного трафика;
   б) повышение безопасности сети;
   в) улучшение управляемости сети;
   г) повышение производительности сети.
5. Должен ли алгоритм покрывающего дерева учитывать наличие в сети VLAN?

Часть IV

Сети TCP/IP

- Глава 14. Адресация в стеке протоколов TCP/IP
- Глава 15. Протокол межсетевого взаимодействия
- Глава 16. Протоколы транспортного уровня TCP и UDP
- Глава 17. Протоколы маршрутизации

Эта часть книги посвящена самой популярной сетевой технологии, которая, появившись более 40 лет назад как результат создания Интернета, сегодня используется практически во всех существующих и вновь создаваемых сетях.

Прежде чем перейти к рассмотрению стека TCP/IP, давайте подведем некоторые итоги, вспомним, что мы уже изучили в первых трех частях, и поговорим о том, с чем нам еще предстоит познакомиться. В части I мы обсудили на концептуальном уровне большинство проблем, которым посвящен этот учебник. Возможно, это самая сложная и важная часть книги — ведь от того, насколько хорошо заложен фундамент, зависит прочность основанных на нем знаний. Мы не раз обращались и будем обращаться к материалам части I в дальнейшем.

Части II и III посвящены конкретным технологиям передачи данных соответственно физического и канального уровней. В них существенно реже описывались абстрактные модели сети в виде графа или «облака», в котором «плавают» компьютеры. Вместо этого на первый план вышли конкретные протоколы, форматы кадров и реальное оборудование.

Что же ждет читателя в этой части — части IV? Следуя логике, диктуемой моделью OSI, вслед за частями, в которых были изучены технологии физического и канального уровней, мы рассмотрим в части IV средства сетевого уровня, то есть средства, обеспечивающие возможность объединения множества разных сетей в единую сеть. Учитывая, что бесспорным лидером среди протоколов сетевого уровня является протокол IP, мы будем рассматривать вопросы построения объединенных сетей на его примере. При этом мы дадим по возможности широкую картину взаимодействия всех протоколов стека TCP/IP. Следуя важной современной тенденции все более широкого использования усовершенствованной версии протокола IPv6, мы уделим внимание его коренным образом переработанным системе адресации и маршрутизации, а также тем возможностям, которые предлагаются для плавного перехода на эту новую версию.

Забегая вперед, хотим предупредить читателя, что в следующих частях книги мы еще не раз обратимся к стеку TCP/IP. В части V «Технологи глобальных сетей» изучаются особенности работы протокола IP «поверх» сетей, поддерживающих технику виртуальных каналов, а также тесно связанная с IP технология MPLS.

Часть VI «Сетевые информационные службы» в значительной мере посвящена прикладным протоколам стека TCP/IP: HTTP (веб-служба), SMTP, IMAP и POP (почтовая служба), SNMP и telnet (системы управления сетью).

Материал последней части VII, «Безопасность компьютерных сетей», также тесно связан с протоколами и технологиями IP-сетей. Здесь рассматриваются уязвимости и методы защиты транспортных протоколов и информационных служб IP-сетей, фильтрация IP-трафика, протокол трансляции IP-адресов NAT, защищенная версия протокола IP — протокол IPSec.

ГЛАВА 14 Адресация в стеке протоколов TCP/IP

Важную часть технологии TCP/IP составляют задачи адресации, к числу которых относятся следующие:

- *Согласованное использование адресов различного типа.* Эта задача включает отображение адресов разных типов друг на друга, например сетевого IP-адреса на локальный, доменного имени — на IP-адрес.
- *Обеспечение уникальности адресов.* В зависимости от типа адреса требуется обеспечивать однозначность адресации в пределах компьютера, подсети, корпоративной сети или Интернета.
- *Конфигурирование сетевых интерфейсов и сетевых приложений.*

Каждая из перечисленных задач имеет достаточно простое решение для сети, число узлов которой не превосходит нескольких десятков. Например, для отображения символьного доменного имени на IP-адрес достаточно поддерживать на каждом хосте таблицу всех символьных имен, используемых в сети, и соответствующих им IP-адресов. Столь же просто «вручную» присвоить всем интерфейсам в небольшой сети уникальные адреса. Однако в крупных сетях эти же задачи усложняются настолько, что требуют принципиально иных решений.

Ключевым словом, которое характеризует принятый в TCP/IP подход к решению этих проблем, является **масштабируемость**. Процедуры, предлагаемые TCP/IP для назначения, отображения и конфигурирования адресов, одинаково хорошо работают в сетях разного масштаба. В этой главе наряду с собственно схемой образования IP-адресов мы познакомимся с наиболее популярными масштабируемыми средствами поддержки адресации в сетях TCP/IP: технологией бесклассовой междоменной маршрутизации, системой доменных имен, протоколом динамического конфигурирования хостов.

В этой главе речь идет о системе адресации, используемой в четвертой версии протокола **IPv4**, которая существенно отличается от системы адресации в версии IPv6. Особенности **IPv6** рассматриваются в главе 15.

Структура стека протоколов TCP/IP

Сегодня стек TCP/IP широко используется как в глобальных, так и в локальных сетях. Этот стек имеет иерархическую структуру, в которой определено четыре уровня (рис. 14.1).

Прикладной уровень стека TCP/IP соответствует трем верхним уровням модели OSI: прикладному, представления и сеансовому. Он объединяет сервисы, предоставляемые системой пользовательским приложениям. За долгие годы применения в сетях различных стран и организаций стек TCP/IP накопил большое количество протоколов и служб прикладного уровня. К ним относятся такие распространенные протоколы, как протокол передачи файлов (File Transfer Protocol, FTP), протокол эмуляции терминала telnet, простой протокол передачи почты (Simple Mail Transfer Protocol, SMTP), протокол пере-

Прикладной уровень	FTP, Telnet, HTTP, SMTP, SNMP, TFTP
Транспортный уровень	TCP, UDP
Сетевой уровень	IP, ICMP, RIP, OSPF
Уровень сетевых интерфейсов	Не регламентируется

Рис. 14.1. Иерархическая структура стека TCP/IP

дачи гипертекста (Hypertext Transfer Protocol, HTTP) и многие другие. Протоколы прикладного уровня развертываются на хостах[1].

Транспортный уровень стека TCP/IP может предоставлять вышележащему уровню два типа сервиса:

- гарантированную доставку обеспечивает **протокол управления передачей** (Transmission Control Protocol, **TCP**);
- доставку по возможности, или с максимальными усилиями, обеспечивает **протокол пользовательских дейтаграмм** (User Datagram Protocol, **UDP**).

Для того чтобы обеспечить надежную доставку данных, протокол TCP предусматривает установление логического соединения, что позволяет ему нумеровать пакеты, подтверждать их прием квитанциями, в случае потери организовывать повторные передачи, распознавать и уничтожать дубликаты, доставлять прикладному уровню пакеты в том порядке, в котором они были отправлены. Благодаря этому протоколу объекты на хосте-отправителе и хосте-получателе могут поддерживать обмен данными в дуплексном режиме. TCP дает возможность без ошибок доставить сформированный на одном из компьютеров поток байтов на любой другой компьютер, входящий в составную сеть.

Второй протокол этого уровня, UDP, является простейшим дейтаграммным протоколом, который используется тогда, когда задача надежного обмена данными либо вообще не ставится, либо решается средствами более высокого уровня — прикладным уровнем или пользовательскими приложениями.

В функции протоколов TCP и UDP входит также исполнение роли связующего звена между прилегающими к транспортному уровню прикладным и сетевым уровнями. От прикладного протокола транспортный уровень принимает задание на передачу данных с тем или иным качеством прикладному уровню-получателю. Нижележащий сетевой уровень протоколы TCP и UDP рассматривают как своего рода инструмент, не очень надежный, но способный перемещать пакет в свободном и рискованном путешествии по составной сети.

---

[1] В Интернете (а значит, и в стеке протоколов TCP/IP) конечный узел традиционно называют *хостом*, а маршрутизатор — *шлюзом*. Далее мы будем использовать пары терминов «конечный узел» — «хост» и «маршрутизатор» — «шлюз» как синонимы, чтобы отдать дань уважения традиционной терминологии Интернета и в то же время не отказываться от современных терминов.

Программные модули, реализующие протоколы TCP и UDP, подобно модулям протоколов прикладного уровня устанавливаются на хостах.

Сетевой уровень, называемый также **уровнем интернета**, является стержнем всей архитектуры TCP/IP. Именно этот уровень, функции которого соответствуют сетевому уровню модели OSI, обеспечивает перемещение пакетов в пределах составной сети, образованной объединением нескольких подсетей. Протоколы сетевого уровня поддерживают интерфейс с вышележащим транспортным уровнем, получая от него запросы на передачу данных по составной сети, а также с нижележащим уровнем сетевых интерфейсов, о функциях которого мы расскажем далее.

Основным протоколом сетевого уровня является межсетевой протокол (Internet Protocol, **IP**). В его задачу входит продвижение пакета между сетями — от одного маршрутизатора к другому до тех пор, пока пакет не попадет в сеть назначения. В отличие от протоколов прикладного и транспортного уровней протокол IP развертывается не только на хостах, но и на всех маршрутизаторах (шлюзах). Протокол IP — это дейтаграммный протокол, работающий без установления соединений по принципу доставки с максимальными усилиями. Такой тип сетевого сервиса называют также «ненадежным».

К сетевому уровню TCP/IP часто относят протоколы, выполняющие вспомогательные функции по отношению к IP. Это прежде всего протоколы маршрутизации RIP и OSPF, предназначенные для изучения топологии сети, определения маршрутов и составления таблиц маршрутизации, на основании которых протокол IP перемещает пакеты в нужном направлении. По этой же причине к сетевому уровню могут быть отнесены протокол межсетевых управляющих сообщений (Internet Control Message Protocol, **ICMP**), предназначенный для передачи маршрутизатором источнику сведений об ошибках, возникших при передаче пакета, и некоторые другие протоколы.

Идеологическим отличием архитектуры стека TCP/IP от многоуровневой архитектуры других стеков является интерпретация функций самого нижнего уровня — **уровня сетевых интерфейсов**.

Напомним, что нижние уровни модели OSI (канальный и физический) реализуют множество функций: доступа к среде передачи, формирования кадров, согласования величин электрических сигналов, кодирования и синхронизации, а также некоторые другие. Все эти весьма конкретные функции составляют суть таких протоколов обмена данными, как Ethernet, PPP и многих других.

У нижнего уровня стека TCP/IP задача существенно проще — он отвечает только за организацию взаимодействия с подсетями разных технологий, входящими в составную сеть. TCP/IP рассматривает любую подсеть, входящую в составную сеть, как средство транспортировки пакетов между двумя соседними маршрутизаторами.

Задачу организации интерфейса между технологией TCP/IP и любой другой технологией промежуточной сети упрощенно можно свести к двум задачам:

- упаковка (инкапсуляция) IP-пакета в единицу передаваемых данных промежуточной сети;
- преобразование сетевых адресов в адреса технологии данной промежуточной сети.

Такой гибкий подход упрощает решение проблемы расширения набора поддерживаемых технологий. При появлении новой популярной технологии она быстро включается в стек TCP/IP путем разработки соответствующего стандарта, определяющего метод инкапсуляции IP-пакетов в ее кадры (например, спецификация RFC 1577, определяющая работу протокола IP через сети ATM, появилась в 1994 году вскоре после принятия основных

стандартов ATM). Так как для каждой вновь появляющейся технологии разрабатываются собственные интерфейсные средства, функции этого уровня нельзя определить раз и навсегда, именно поэтому нижний уровень стека TCP/IP не регламентируется.

Каждый коммуникационный протокол оперирует некоторой единицей передаваемых данных. Названия этих единиц иногда закрепляются стандартом, а чаще просто определяются традицией. В стеке TCP/IP за многие годы его существования образовалась устоявшаяся терминология в этой области (рис. 14.2).

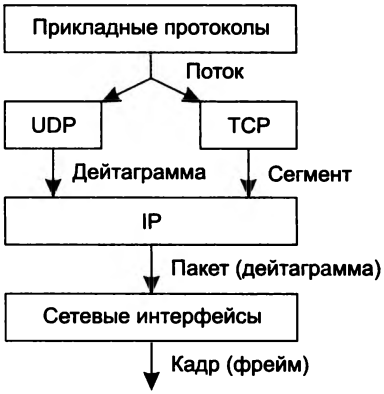


Рис. 14.2. Названия протокольных единиц данных в TCP/IP

Потоком данных, **информационным потоком**, или просто **потоком**, называют данные, поступающие от приложений на вход протоколов транспортного уровня — TCP и UDP.

Протокол TCP «нарезает» из потока данных **сегменты**.

Единицу данных протокола UDP часто называют **дейтаграммой**, или **датаграммой**. Дейтаграмма — это общее название для единиц данных, которыми оперируют протоколы без установления соединений. К таким протоколам относится и протокол IP, поэтому его единицу данных иногда тоже называют дейтаграммой, хотя достаточно часто используется и другой термин — **пакет**.

В стеке TCP/IP единицы данных любых технологий, в которые упаковываются IP-пакеты для их последующей передачи через сети составной сети, принято называть также **кадрами**, или **фреймами**. При этом не имеет значения, какое название используется для этой единицы данных в технологии составляющей сети. Для TCP/IP фреймом является и кадр Ethernet, и ячейка ATM, и пакет X.25 в тех случаях, когда они выступают в качестве контейнера, в котором IP-пакет переносится через составную сеть.

Типы адресов стека TCP/IP

Для идентификации сетевых интерфейсов используются три типа адресов:

- локальные (аппаратные) адреса;
- сетевые адреса (IP-адреса);
- символьные (доменные) имена.

Локальные адреса

В большинстве технологий LAN (Ethernet, FDDI, Token Ring) для однозначной адресации интерфейсов используются **MAC-адреса**. Существует немало технологий (X.25, ATM, frame relay), в которых применяются другие схемы адресации. Роль, которую играют эти адреса в TCP/IP, не зависит от того, какая именно технология используется в подсети, поэтому все они имеют общее название — **локальные (аппаратные) адреса**.

Слово «локальный» в контексте TCP/IP означает «действующий не во всей составной сети, а лишь в пределах подсети». Именно в таком смысле понимаются здесь термины: «локальная технология» (технология, на основе которой построена подсеть) и «локальный адрес» (адрес, который используется некоторой локальной технологией для адресации узлов в пределах подсети). Напомним, что в качестве подсети («локальной сети» в терминологии TCP/IP) может выступать сеть, построенная как на основе технологии LAN, например Ethernet, FDDI, так и на основе технологии WAN, например X.25, Frame Relay. Следовательно, говоря о подсети, мы используем слово «локальная» не как характеристику технологии, на которой построена эта подсеть, а как указание на роль, которую играет эта подсеть в архитектуре составной сети.

Сложности могут возникнуть и при интерпретации определения «аппаратный». В данном случае термин «аппаратный» подчеркивает концептуальное представление разработчиков стека TCP/IP о подсети как о некотором вспомогательном *аппаратном* средстве, единственной функцией которого является перемещение IP-пакета через подсеть до ближайшего шлюза (маршрутизатора). И не важно, что реально нижележащая локальная технология может быть достаточно сложной, все ее сложности технологией TCP/IP игнорируются.

Рассмотрим, например, случай (в настоящее время представляющий поучительный исторический пример), когда в составную сеть TCP/IP входит сеть IPX/SPX. Последняя сама может быть разделена на подсети, и так же как IP-сеть, она идентифицирует свои узлы аппаратными и сетевыми IPX-адресами. Но технология TCP/IP игнорирует многоуровневое строение сети IPX/SPX и рассматривает в качестве локальных адресов узлов подсети IPX/SPX адреса сетевого уровня данной технологии (IPX-адреса). Аналогично если в составную сеть включена сеть X.25, то локальными адресами узлов этой сети для протокола IP будут соответственно адреса X.25.

Сетевые IP-адреса

Чтобы технология TCP/IP могла решать свою задачу объединения сетей, ей необходима собственная глобальная система адресации, *не зависящая от способов адресации узлов в отдельных сетях*. Эта система адресации должна позволять универсальным и однозначным способом идентифицировать любой интерфейс составной сети. Очевидным решением является уникальная нумерация всех сетей составной сети, а затем нумерация всех узлов в пределах каждой из этих сетей. Пара, состоящая из **номера сети** и **номера узла**, отвечает поставленным условиям и может являться **сетевым адресом**, или в терминологии TCP/IP — **IP-адресом**.

Глядя на топологическую схему IP-сети, можно отметить, что маршрутизатор по определению входит сразу в несколько сетей, следовательно, каждый его интерфейс должен иметь собственный IP-адрес. Конечный узел, имеющий несколько сетевых интерфейсов, также может входить в несколько IP-сетей, а значит, иметь несколько IP-адресов — по

числу сетевых связей. Таким образом, подчеркнем еще раз — IP-адрес идентифицирует не отдельный узел сети (компьютер или маршрутизатор), а одно сетевое соединение, или, что одно и то же в данном контексте, один сетевой интерфейс[1].

Каждый раз, когда пакет направляется адресату через составную сеть, в его заголовке указывается IP-адрес узла назначения. По номеру сети назначения каждый очередной маршрутизатор находит IP-адрес следующего маршрутизатора. Перед тем как отправить пакет в следующую сеть, маршрутизатор должен определить на основании найденного IP-адреса следующего маршрутизатора его локальный адрес. Между IP-адресом и локальным адресом узла нет никакой функциональной зависимости, поэтому единственный способ установления соответствия — ведение таблицы. Эту задачу решает протокол разрешения адресов ARP (рис. 14.3).

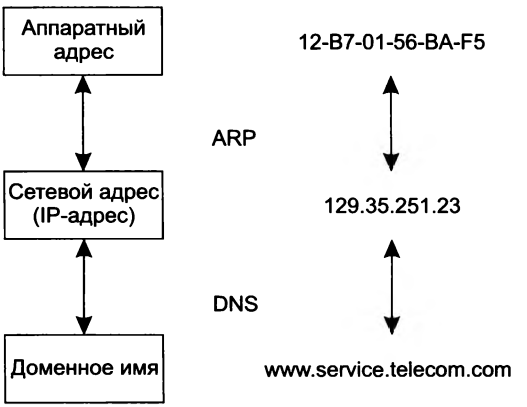


Рис. 14.3. Преобразование адресов

Доменные имена

Для идентификации компьютеров аппаратное и программное обеспечение в сетях TCP/IP полагается на IP-адреса. Например, команда ftp://192.45.66.17 будет устанавливать сеанс связи с нужным ftp-сервером, а команда http://203.23.106.33 откроет начальную страницу на корпоративном веб-сервере. Однако пользователи обычно предпочитают работать с более удобными **символьными именами** компьютеров.

Символьные идентификаторы сетевых интерфейсов в пределах составной сети строятся по иерархическому принципу. Составляющие полного символьного (или доменного) имени в IP-сетях разделяются точкой и перечисляются в следующем порядке: сначала — простое имя хоста, затем — имя группы хостов (например, имя организации), потом — имя более крупной группы (**домена**), и так — до имени домена самого высокого уровня (например, домена, объединяющего организации по географическому принципу: RU — Россия, UK — Великобритания, US — США). Примером доменного имени может служить имя base2.sales.zil.ru.

[1] Тем не менее нами часто будет использоваться хотя и не совсем корректное, но зато более упрощенное выражение «адрес узла».

Символьные имена называют также **доменными именами**.

Между IP-адресом узла и его доменным именем (так же, как и локальным адресом) нет никакой функциональной зависимости, поэтому для установления соответствия требуются таблицы. В сетях TCP/IP используется специальная сетевая служба, называемая **системой доменных имен** (Domain Name System, **DNS**), которая автоматически устанавливает соответствие между доменными именами и IP-адресами на основании создаваемых администраторами сети таблиц соответствия. По этой причине доменные имена называют также **DNS-именами**.

В общем случае один сетевой интерфейс может иметь несколько локальных адресов, сетевых адресов и доменных имен.

Формат IP-адреса

В заголовке IP-пакета предусмотрены поля для хранения IP-адреса отправителя и IP-адреса получателя. Каждое из этих полей имеет фиксированную длину 4 байта (32 бита). Как уже было сказано, IP-адрес состоит из двух логических частей — номера сети и номера узла в сети.

Наиболее распространенной формой представления IP-адреса является запись в виде четырех чисел, представляющих значения каждого байта в десятичной форме и разделенных точками, например:

128.10.2.30

Этот же адрес может быть представлен в двоичном формате:

10000000 00001010 00000010 00011110

А также в шестнадцатеричном формате:

80.0A.02.1D

Заметим, что запись адреса не предусматривает *специального разграничительного знака* между номером сети и номером узла. Вместе с тем при передаче пакета по сети часто возникает необходимость разделить адрес на эти две части. Например, маршрутизация, как правило, осуществляется на основании номера сети, поэтому каждый маршрутизатор, получая пакет, должен прочитать из соответствующего поля заголовка адрес назначения и выделить из него номер сети. Каким образом маршрутизаторы определяют, какая часть из 32 бит, отведенных под IP-адрес, относится к номеру сети, а какая — к номеру узла?

Можно предложить несколько вариантов решения этой проблемы.

- Простейший из них состоит в использовании **фиксированной границы**. При этом все 32-битное поле адреса заранее делится на две части не обязательно равной, но фиксированной длины, в одной из которых всегда будет размещаться номер сети, в другой — номер узла. Решение очень простое, но хорошее ли? Поскольку поле, которое отводится для хранения номера узла, имеет фиксированную длину, все сети будут иметь одинаковое максимальное число узлов. Если, например, под номер сети отвести один первый байт, то все адресное пространство распадется на сравнительно небольшое (2^8) число сетей огромного размера (2^{24} узлов). Если границу передвинуть дальше вправо, то сетей станет больше, но все равно все они будут одинакового размера. Очевидно, что такой жесткий подход не позволяет дифференцированно удовлетворять потребности

отдельных предприятий и организаций. Именно поэтому он не нашел применения, хотя и использовался на начальном этапе существования технологии TCP/IP.

- Второй подход основан на применении *маски*, которая позволяет максимально гибко устанавливать границу между номером сети и номером узла. При таком подходе адресное пространство можно использовать для создания множества сетей разного размера.

  Маска — это число, применяемое в паре с IP-адресом, причем двоичная запись маски содержит непрерывную последовательность единиц в тех разрядах, которые должны в IP-адресе интерпретироваться как номер сети. Граница между последовательностями единиц и нулей в маске соответствует границе между номером сети и номером узла в IP-адресе. Например, если маска, связываемая с некоторым IP-адресом, имеет вид 11111111111100000000000000000000, то номеру сети соответствуют 10 старших разрядов в двоичном представлении данного IP-адреса.

- И наконец, способ, основанный на **классах адресов**. Этот способ представляет собой компромисс по отношению к двум предыдущим: размеры сетей хотя и не могут быть произвольными, как при использовании масок, но и не должны быть одинаковыми, как при установлении фиксированных границ. Вводится пять классов адресов: A, B, C, D, E. Три из них — A, B и C — предназначены для адресации сетей, а два — D и E — имеют специальное назначение. Для каждого класса сетевых адресов определено собственное положение границы между номером сети и номером узла.

Классы IP-адресов

Признаком, на основании которого IP-адрес относят к тому или иному классу, являются значения нескольких первых битов адреса (рис. 14.4).

Рис. 14.4. Классы IP-адресов

В табл. 14.1 приведены диапазоны адресов и максимальное число сетей и узлов, соответствующих каждому классу.

Таблица 14.1. Классы IP-адресов

Класс	Первые биты	Наименьший номер сети	Наибольший номер сети	Максимальное число узлов в сети
A	0	1.0.0.0 (0 — не используется)	126.0.0.0 (127 — зарезервирован)	2^{24}, поле 3 байта
B	10	128.0.0.0	191.255.0.0	2^{16}, поле 2 байта
C	110	192.0.0.0	223.255.255.0	2^{8}, поле 1 байт
D	1110	224.0.0.0	239.255.255.255	Групповые адреса
E	11110	240.0.0.0	247.255.255.255	Зарезервировано

Исходя из приведенной структуры адресов и информации из таблицы, можно сделать несколько очевидных выводов. Сетей класса A сравнительно немного, зато количество узлов в них очень большое, оно может достигать 2^{24}, что равно 16 777 216 узлов. Сетей класса B больше, чем сетей класса A, но их размеры меньше, максимальное количество узлов в сетях класса B составляет 2^{16} (65 536). Сетей класса C больше всего, но они характеризуются самым маленьким максимально возможным количеством узлов, всего — 2^{8} (256).

В то время как адреса классов A, B и C используются для идентификации отдельных сетевых интерфейсов, то есть являются **индивидуальными адресами** (unicast address), **групповые адреса** (multicast address) класса D идентифицируют группу сетевых интерфейсов, которые в общем случае могут принадлежать разным сетям. Интерфейс, входящий в группу, получает наряду с обычным индивидуальным IP-адресом еще один групповой адрес. Если при отправке пакета в качестве адреса назначения указан адрес класса D, то такой пакет должен быть доставлен всем узлам, которые входят в группу. Адрес класса D начинается с последовательности 1110.

Если адрес начинается с последовательности 11110, то это значит, что данный адрес относится к **классу E**. Адреса этого класса зарезервированы для будущих применений.

Чтобы получить из IP-адреса номер сети и номер узла, требуется не только разделить адрес на две соответствующие части, но и дополнить каждую из них нулями до полных четырех байтов. Возьмем, например, адрес класса B 129.64.134.5. Первые два байта идентифицируют сеть, а последующие два — узел. Таким образом, номером сети является адрес 129.64.0.0, а номером узла — адрес 0.0.134.5.

Особые IP-адреса

В TCP/IP существуют ограничения при назначении IP-адресов, а именно номера сетей и номера узлов *не могут состоять из одних двоичных нулей или единиц*. Отсюда следует, что максимальное количество узлов, приведенное в табл. 14.1 для сетей каждого класса, должно быть уменьшено на 2. Например, в адресах класса C под номер узла отводится 8 бит, которые позволяют задать 256 номеров: от 0 до 255. Однако в действительности максимальное

число узлов в сети класса C не может превышать 254, так как адреса 0 и 255 запрещены для адресации сетевых интерфейсов. Из этих же соображений следует, что конечный узел не может иметь адрес типа 98.255.255.255, поскольку номер узла в этом адресе класса A состоит из одних двоичных единиц.

Введя эти ограничения, разработчики технологии TCP/IP получили возможность расширить функциональность системы адресации следующим образом:

- Если IP-адрес состоит только из двоичных нулей, то он называется **неопределенным адресом** и обозначает адрес того узла, который сгенерировал этот пакет. Адрес такого вида в особых случаях помещается в заголовок IP-пакета в поле адреса отправителя.
- Если в поле номера сети стоят только нули, то по умолчанию считается, что узел назначения принадлежит той же самой сети, что и узел, который отправил пакет. Такой адрес также может быть использован только в качестве адреса отправителя.
- Если все двоичные разряды IP-адреса равны 1, то пакет с таким адресом назначения должен рассылаться всем узлам, находящимся в той же сети, что и источник этого пакета. Такой адрес называется **ограниченным широковещательным** (limited broadcast). Ограниченность в данном случае означает, что пакет не выйдет за границы данной подсети ни при каких условиях.
- Если в поле адреса назначения в разрядах, соответствующих номеру узла, стоят только единицы, то пакет, имеющий такой адрес, рассылается *всем* узлам сети, номер которой указан в адресе назначения. Например, пакет с адресом 192.190.21.255 будет направлен всем узлам сети 192.190.21.0. Такой тип адреса называется **широковещательным** (broadcast).

ВНИМАНИЕ

В протоколе IP нет понятия широковещания в том смысле, в котором оно используется в протоколах канального уровня локальных сетей, когда данные должны быть доставлены абсолютно всем узлам сети. Как ограниченный, так и обычный вариант широковещательной рассылки имеет пределы распространения в составной сети: они ограничены либо сетью, которой принадлежит источник пакета, либо сетью, номер которой указан в адресе назначения. Деление сети с помощью маршрутизаторов на части локализует широковещательный шторм пределами одной из подсетей просто потому, что нет способа адресовать пакет одновременно всем узлам всех сетей составной сети.

Особый смысл имеет IP-адрес, первый октет которого равен 127. Этот адрес является *внутренним адресом стека протоколов* компьютера (или маршрутизатора). Он используется для тестирования программ, а также для организации работы клиентской и серверной частей приложения, установленных на одном компьютере. Обе программные части данного приложения спроектированы в расчете на то, что они будут обмениваться сообщениями по сети. Но какой же IP-адрес они должны использовать для этого? Адрес сетевого интерфейса компьютера, на котором они установлены? Но это приводит к избыточным передачам пакетов в сеть. Экономичным решением является применение внутреннего адреса 127.0.0.0. В IP-сети запрещается присваивать сетевым интерфейсам IP-адреса, начинающиеся со значения 127. Когда программа посылает данные по IP-адресу 127.x.x.x, то данные не передаются в сеть, а возвращаются модулям верхнего уровня того же компьютера как только что принятые. Маршрут перемещения данных образует «петлю», поэтому этот адрес называется **адресом обратной петли** (loopback).

Уже упоминавшиеся *групповые адреса*, относящиеся к классу D, предназначены для экономичного распространения в Интернете или большой корпоративной сети аудио- или видеопрограмм, адресованных сразу большой аудитории слушателей или зрителей. Если групповой адрес помещен в поле адреса назначения IP-пакета, то данный пакет должен быть доставлен сразу нескольким узлам, которые образуют группу с номером, указанным в поле адреса. Один и тот же узел может входить в несколько групп. В общем случае члены группы могут распределяться по различным сетям, находящимся друг от друга на произвольно большом расстоянии. Групповой адрес не делится на номера сети и узла и обрабатывается маршрутизатором особым образом. Основное назначение групповых адресов — распространение информации по схеме «один ко многим». От того, найдут ли групповые адреса широкое применение (сейчас их используют в основном небольшие экспериментальные «островки» в Интернете), зависит, сможет ли Интернет создать серьезную конкуренцию радио и телевидению.

Использование масок при IP-адресации

Снабжая каждый IP-адрес маской, можно отказаться от понятий классов адресов и сделать систему адресации более гибкой.

Пусть, например, для IP-адреса 129.64.134.5 указана маска 255.255.128.0, то есть в двоичном виде IP-адрес 129.64.134.5 равен:

10000001.01000000.10000110.00000101,

В то время как маска 255.255.128.0 выглядит так:

11111111.11111111.10000000.00000000.

Если игнорировать маску и интерпретировать адрес 129.64.134.5 на основе классов, то номером сети является 129.64.0.0, а номером узла — 0.0.134.5 (поскольку адрес относится к классу B).

Если же использовать маску, то 17 последовательных двоичных единиц в маске 255.255.128.0, «наложенные» на IP-адрес 129.64.134.5, делят его на две части:

- номер сети: 10000001.01000000.1;
- номер узла: 0000110.00000101.

В десятичной форме записи номера сети и узла, дополненные нулями до 32 бит, выглядят соответственно как 129.64.128.0 и 0.0.6.5.

Наложение маски можно интерпретировать как выполнение логической операции И (AND). Так, в предыдущем примере номер сети из адреса 129.64.134.5 является результатом выполнения логической операции AND с маской 255.255.128.0:

10000001 01000000 10000110 00000101

AND

11111111.11111111.10000000.00000000

Для записи масок используются и другие форматы. Например, удобно интерпретировать значение маски, записанной в шестнадцатеричном коде: FF.FF.00.00 — маска для адресов класса B. Еще чаще встречается запись с префиксом 185.23.44.206/26 — данная запись говорит о том, что маска для этого адреса содержит 26 единиц. В табл. 15.2 приведены маски для стандартных классов сетей, записанные в разных форматах.

Таблица 14.2. Маски для стандартных классов сетей

Класс адресов	Десятичная форма	Двоичная форма	Шестнадцатеричная форма	Префикс
A	255.0.0.0	11111111. 00000000. 00000000. 00000000	FF.00.00.00	/8
B	255.255.0.0	11111111. 11111111. 00000000. 00000000	FF.FF.00.00	/16
C	255.255.255.0	11111111. 11111111. 11111111. 00000000	FF.FF.FF.00	/24

Механизм масок широко распространен в маршрутизации IP, причем маски могут использоваться для самых разных целей. С их помощью администратор может разбить одну выделенную ему поставщиком услуг сеть определенного класса на несколько других, не требуя от него дополнительных номеров сетей, — эта операция называется *разделением на подсети* (subnetting). На основе этого же механизма поставщики услуг могут объединять адресные пространства нескольких сетей путем введения так называемых «префиксов» с целью уменьшения объема таблиц маршрутизации и повышения за счет этого производительности маршрутизаторов — такая операция называется *объединением подсетей* (supernetting). Подробнее об этом мы поговорим при изучении технологии бесклассовой междоменной маршрутизации.

Порядок назначения IP-адресов

По определению схема IP-адресации должна обеспечивать уникальность нумерации сетей, а также уникальность нумерации узлов в пределах каждой из сетей. Следовательно, процедуры назначения номеров как сетям, так и узлам сетей должны быть *централизованными*. Рекомендуемый порядок назначения IP-адресов дается в спецификации RFC 2050.

Назначение адресов автономной сети

Когда дело касается сети, являющейся частью Интернета, уникальность нумерации может быть обеспечена только усилиями специально созданных для этого центральных органов. В небольшой же автономной IP-сети условие уникальности номеров сетей и узлов может быть выполнено «вручную» сетевым администратором.

В этом случае в распоряжении администратора имеется все адресное пространство, так как совпадение IP-адресов в не связанных между собой сетях не вызовет никаких отрицательных последствий. Администратор может выбирать адреса произвольным образом, соблюдая лишь синтаксические правила и учитывая ограничения на особые адреса.

Однако при таком подходе исключена возможность в будущем подсоединить данную сеть к Интернету. Действительно, произвольно выбранные адреса данной сети могут совпасть с централизовано назначенными адресами Интернета. Для того чтобы избежать коллизий, связанных с такого рода совпадениями, в стандартах Интернета определено несколько диапазонов так называемых **частных адресов**, рекомендуемых для автономного использования:

- в классе A — сеть 10.0.0.0;
- в классе B — диапазон из 16 номеров сетей (172.16.0.0–172.31.0.0);
- в классе C — диапазон из 255 сетей (192.168.0.0–192.168.255.0).

Эти адреса, исключенные из множества централизованно распределяемых, составляют огромное адресное пространство, достаточное для нумерации узлов автономных сетей практически любых размеров. Заметим также, что частные адреса, как и при произвольном выборе адресов, в разных автономных сетях могут совпадать. В то же время использование частных адресов для адресации автономных сетей делает возможным их корректное подключение к Интернету. Применяемые при этом специальные технологии подключения исключают коллизии адресов[1].

Централизованное распределение адресов

В больших сетях, подобных Интернету, уникальность сетевых адресов гарантируется централизованной иерархически организованной системой их распределения. Номер сети может быть назначен только по рекомендации специального подразделения Интернета. Главным органом регистрации глобальных адресов в Интернете с 1998 года является неправительственная некоммерческая организация **ICANN** (Internet Corporation for Assigned Names and Numbers). Эта организация координирует работу региональных отделов, деятельность которых охватывает большие географические площади: ARIN (Америка), RIPE (Европа), APNIC (Азия и Тихоокеанский регион). Региональные отделы выделяют блоки адресов сетей крупным поставщикам услуг, а те, в свою очередь, распределяют их между своими клиентами, среди которых могут быть и более мелкие поставщики.

Проблемой централизованного распределения адресов является их дефицит. Уже сравнительно давно очень трудно получить адрес класса B и практически невозможно стать обладателем адреса класса A. При этом надо отметить, что дефицит обусловлен не только ростом сетей, но и тем, что имеющееся адресное пространство используется нерационально. Очень часто владельцы сетей класса C расходуют лишь небольшую часть из имеющихся у них 254 адресов. Рассмотрим пример, когда две сети необходимо соединить глобальной связью. В таких случаях в качестве линии связи используют два маршрутизатора, соединенных по двухточечной схеме (рис. 14.5). Для вырожденной сети, образованной линией связи, связывающей порты двух смежных маршрутизаторов, приходится выделять отдельный номер сети, хотя в этой сети всего два узла.

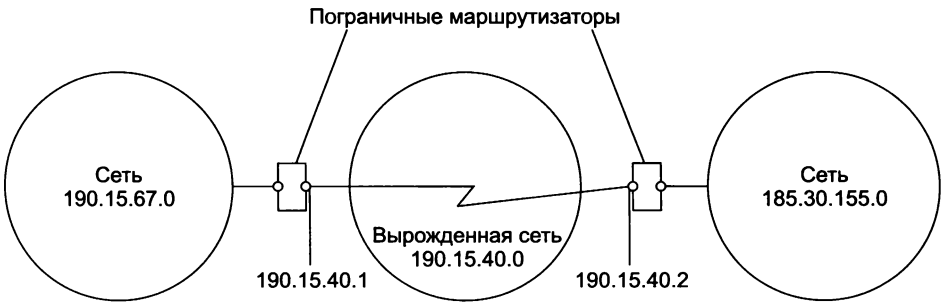


Рис. 14.5. Нерациональное использование пространства IP-адресов

[1] Например, такой технологией является NAT.

Для смягчения проблемы дефицита адресов разработчики стека TCP/IP предлагают разные подходы. Принципиальным решением является переход на новую версию протокола IP — протокол IPv6, в котором резко расширяется адресное пространство. Однако и текущая версия протокола IP (IPv4) поддерживает технологии, направленные на более экономное расходование IP-адресов, такие, например, как NAT и CIDR.

Адресация и технология CIDR

Технология бесклассовой междоменной маршрутизации (Classless Inter-Domain Routing, CIDR) основана на использовании масок для более гибкого распределения адресов и более эффективной маршрутизации. Она допускает произвольное разделение IP-адреса на поля для нумерации сети и узлов. При такой системе адресации клиенту может быть выдан пул адресов, более точно соответствующий его запросу, чем это происходит при адресации, основанной на классах адресов.

Например, если *клиенту А* (рис. 14.6) требуется всего 13 адресов, то вместо выделения ему сети стандартного класса С (класса с наименьшим числом узлов — 256) ему может был назначен пул адресов 193.20.30.0/28. Эта запись, имеющая вид *IP-адрес/маска*, интерпретируется следующим образом: «сеть, не принадлежащая ни к какому стандартному классу, номер которой содержится в 28 старших двоичных разрядах IP-адреса 193.20.30.0, имеющая 4-битовое поле для нумерации 16 узлов». Все это вполне удовлетворяет требованиям клиента А. Очевидно, что такой вариант намного более экономичен, чем раздача сетей стандартных классов «целиком».

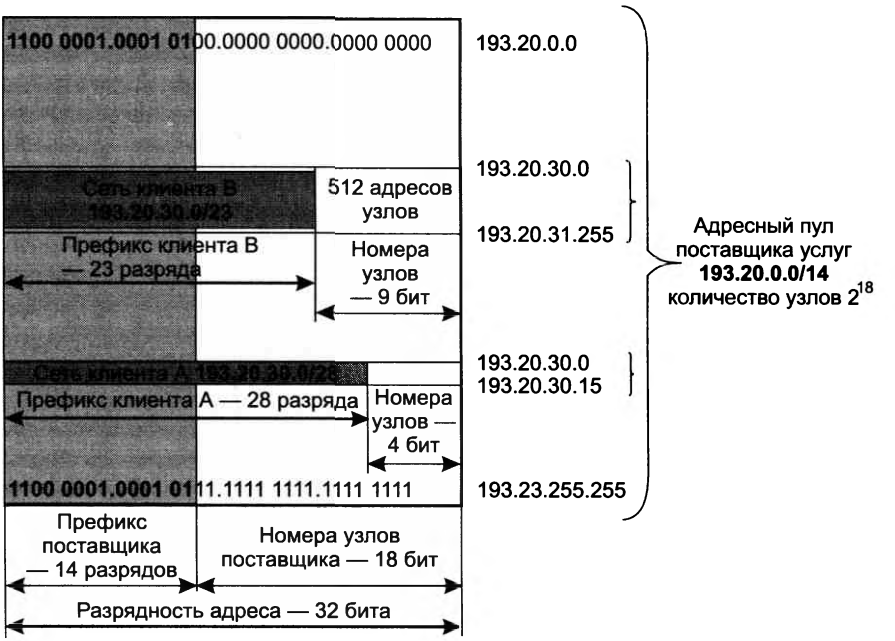


Рис. 14.6. Схема распределения адресного пространства в соответствии с CIDR

Определение пула адресов в виде пары IP-адрес/маска возможно только при выполнении нескольких условий. Прежде всего адресное пространство, из которого организация, распределяющая адреса, «нарезает» адресные пулы для заказчиков, должно быть *непрерывным*. При таком условии все адреса имеют общий **префикс** — одинаковую последовательность цифр в старших разрядах адреса.

Пусть, например, как показано на рис. 14.6, провайдер располагает адресами в диапазоне 193.20.0.0–193.23.255.255, или в десятичной записи:

1100 0001.0001 0100.0000 0000.0000 0000–1100 0001.0001 0111.1111 1111.1111 1111.

Здесь префикс провайдера имеет длину 14 разрядов — 1100 0001.0001 01, что можно записать в виде —193.20.0.0/14. Префикс обычно интерпретируется как номер подсети.

Даже если необходимое клиенту адресное пространство может быть обеспечено предоставлением нескольких сетей стандартного класса, предпочтительным считается вариант IP-адрес/маска, так как в этом случае адреса гарантированно образуют непрерывное пространство. Непрерывность адресного пространства является очень важным свойством, непосредственно влияющим на эффективность маршрутизации, о чем рассказывается в разделе «Маршрутизация с использованием масок» главы 15.

Рассмотрим еще один пример. Пусть *клиент B* (см. рис. 14.6) собирается связать в сеть 500 компьютеров. Вместо того чтобы выделять ему две сети класса C по 256 узлов каждая, клиенту назначают пул адресов в виде пары 193.20.30.0/23. Эта запись означает, что клиенту выделена сеть неопределенного класса, в которой под нумерацию узлов отведено 9 младших битов, что, как и в случае двух сетей класса C, позволяет адресовать 512 узлов. Преимущество этого варианта с маской перед вариантом с двумя сетями состоит в том, что в первом случае *непрерывность пула адресов гарантирована*.

Назначение адресов в виде IP-адрес/маска корректно лишь в том случае, если поле для адресации узлов, полученное применением маски к IP-адресу, содержит только одни нули. Например, определение пула адресов в виде 193.20.00.0/12 ошибочно, так как в поле номера сети (в 20 младших битах) содержится не нулевое значение 0100.0000 0000.0000 0000. В то же время префикс может оканчиваться нулями, например определение пула 193.20.0.0/25, в котором префикс имеет значение 1100 0001.0001 0100.0000 0000.0, вполне корректно.

Итак, для обобщенного представления пула адресов в виде IP/*n* справедливы следующие утверждения:

- значением префикса (номера сети) являются *n* старших двоичных разрядов IP-адреса;
- поле для адресации узлов состоит из (32–*n*) младших двоичных разрядов IP-адреса;
- первый по порядку адрес должен состоять только из нулей;
- количество адресов в пуле равно $2^{(32-n)}$.

Благодаря CIDR поставщик услуг получает возможность «нарезать» блоки из выделенного ему адресного пространства в соответствии с действительными требованиями каждого клиента. Мы еще вернемся к CIDR в главе 15, чтобы обсудить, как эта технология помогает не только экономно расходовать адреса, но и более эффективно осуществлять маршрутизацию.

Отображение IP-адресов на локальные адреса

Одной из главных задач, которая ставилась при создании протокола IP, являлось обеспечение совместной согласованной работы в сети, состоящей из подсетей, в общем случае использующих разные сетевые технологии. При перемещении IP-пакета по составной сети взаимодействие технологии TCP/IP с локальными технологиями подсетей происходит многократно. На каждом маршрутизаторе протокол IP определяет, какому следующему маршрутизатору в этой сети надо направить пакет. В результате решения этой задачи протоколу IP становится известен *IP-адрес* интерфейса следующего маршрутизатора (или конечного узла, если эта сеть является сетью назначения). Чтобы локальная технология сети смогла доставить пакет следующему маршрутизатору, необходимо:

- упаковать пакет в кадр соответствующего для данной сети формата (например, Ethernet);
- снабдить данный кадр *локальным адресом* следующего маршрутизатора.

Решением этих задач занимается уровень сетевых интерфейсов стека TCP/IP[1].

Протокол разрешения адресов

Как уже было сказано, никакой функциональной зависимости между локальным адресом и его IP-адресом не существует, следовательно, единственный способ установления соответствия — ведение таблиц. В результате конфигурирования сети каждый интерфейс «знает» свои IP-адрес и локальный адрес, что можно рассматривать как таблицу, состоящую из одной строки. Проблема состоит в том, как организовать обмен имеющейся информацией между узлами сети.

Для определения локального адреса по IP-адресу используется **протокол разрешения адресов** (Address Resolution Protocol, ARP). Протокол разрешения адресов реализуется различным образом в зависимости от того, работает ли в данной сети протокол локальной сети (Ethernet, Token Ring, FDDI) с возможностью широковещания или же какой-либо из протоколов глобальной сети (MPLS, Frame Relay, ATM), которые, как правило, не поддерживают широковещательный доступ.

Рассмотрим работу протокола ARP в локальных сетях с *широковещанием*.

На рис. 14.7 показан фрагмент IP-сети, включающий две сети — Ethernet1 (из трех конечных узлов *A*, *B* и *C*) и Ethernet2 (из двух конечных узлов *D* и *E*). Сети подключены соответственно к интерфейсам 1 и 2 маршрутизатора. Каждый сетевой интерфейс имеет IP-адрес и MAC-адрес. Пусть в какой-то момент IP-модуль узла *C* направляет пакет узлу *D*. Протокол IP узла *C* определил IP-адрес интерфейса следующего маршрутизатора — это IP_1. Теперь, прежде чем упаковать пакет в кадр Ethernet и направить его маршрутизатору, необходимо определить соответствующий MAC-адрес. Для решения этой задачи протокол IP обращается к протоколу ARP. Протокол ARP поддерживает на каждом интерфейсе сетевого адаптера или маршрутизатора отдельную ARP-таблицу, в которой в ходе функционирования сети накапливается информация о соответствии между IP-адресами и MAC-адресами других интерфейсов данной сети. Первоначально, при включении компьютера или маршрутизатора в сеть, все его ARP-таблицы пусты.

---

[1] См. раздел «Стек TCP/IP» в главе 4.

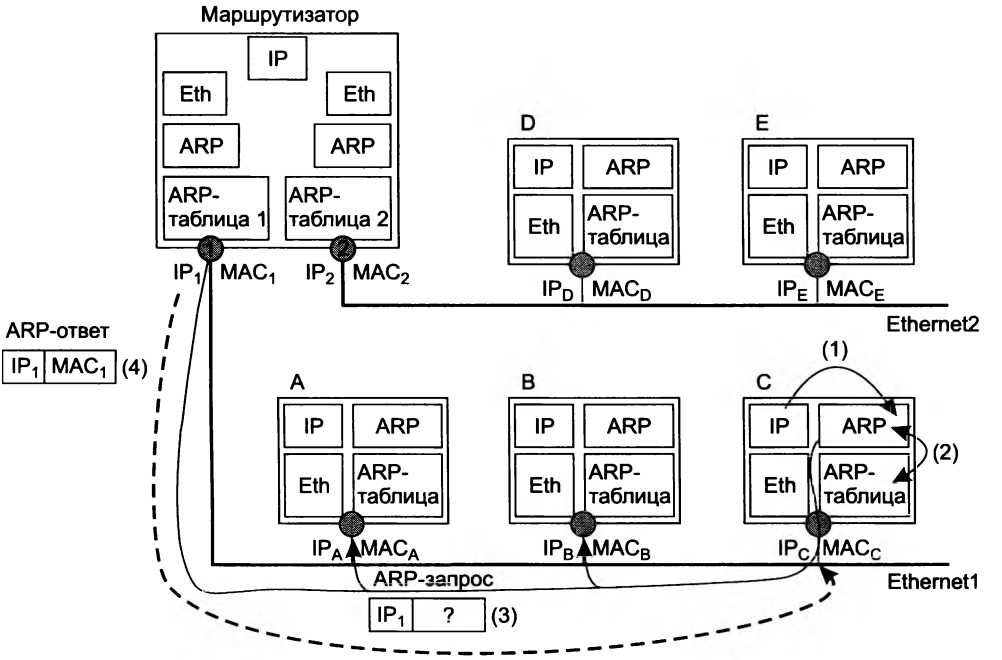


Рис. 14.7. Схема работы протокола ARP

1. На первом шаге происходит передача от протокола IP протоколу ARP примерно такого сообщения: «Какой MAC-адрес имеет интерфейс с адресом IP_1?»
2. Работа протокола ARP начинается с просмотра собственной ARP-таблицы. Предположим, что среди содержащихся в ней записей отсутствует запрашиваемый IP-адрес.
3. В этом случае протокол ARP формирует **ARP-запрос**, вкладывает его в кадр протокола Ethernet и широковещательно рассылает. Заметим, что зона распространения ARP-запроса ограничивается сетью Ethernet1, так как на пути широковещательных кадров барьером стоит маршрутизатор.
4. Все интерфейсы сети Ethernet1 получают ARP-запрос и направляют его «своему» протоколу ARP. ARP сравнивает указанный в запросе адрес IP_1 с IP-адресом собственного интерфейса.
5. Протокол ARP, который констатировал совпадение (в данном случае это ARP интерфейса 1 маршрутизатора), формирует ARP-ответ. В ARP-ответе маршрутизатор указывает локальный адрес MAC_1, соответствующий адресу IP_1 своего интерфейса, и отправляет его запрашивающему узлу (в данном примере узлу C).

На рис. 14.8 показан кадр Ethernet с вложенным в него ARP-сообщением. ARP-запросы и ARP-ответы имеют один и тот же формат. В табл. 14.3 в качестве примера приведены значения полей реального ARP-запроса, переданного по сети Ethernet[1].

[1] Символы 0x означают, что за ними следует число, записанное в шестнадцатеричном формате.

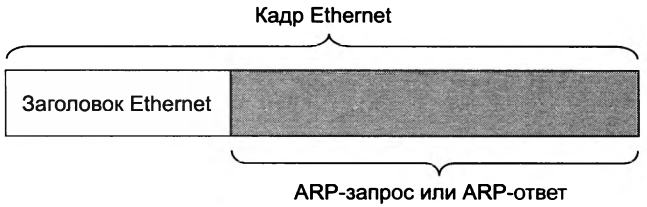


Рис. 14.8. Инкапсуляция ARP-сообщений в кадр Ethernet

Таблица 14.3. Пример ARP-запроса

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)
Длина сетевого адреса	4 (0x4)
Операция	1 (0x1)
Локальный адрес отправителя	008048EB7E60
Сетевой адрес отправителя	194.85.135.75
Локальный (искомый) адрес получателя	000000000000
Сетевой адрес получателя	194.85.135.65

В поле типа сети для сетей Ethernet указывается значение 1. Поле типа протокола позволяет использовать протокол ARP не только с протоколом IP, но и с другими сетевыми протоколами. Для IP значение этого поля равно 0x0800. Длина локального адреса для протокола Ethernet равна 6 байт, а длина IP-адреса — 4 байта. В поле операции для ARP-запросов указывается значение 1, для ARP-ответов — значение 2.

Из этого запроса видно, что в сети Ethernet узел с IP-адресом 194.85.135.75 пытается определить, какой MAC-адрес имеет другой узел той же сети, сетевой адрес которого 194.85.135.65. *Поле искомого локального адреса заполнено нулями.*

Ответ присылает узел, опознавший свой IP-адрес. Если в сети нет машины с искомым IP-адресом, то ARP-ответа не будет. Протокол IP уничтожает IP-пакеты, направляемые по этому адресу. В табл. 14.4 показаны значения полей ARP-ответа, который мог бы поступить на приведенный в табл. 14.3 ARP-запрос.

Таблица 14.4. Пример ARP-ответа

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)

Таблица 14.4 (*продолжение*)

Поле	Значение
Длина сетевого адреса	4 (0x4)
Операция	2 (0x1)
Локальный адрес отправителя	00E0F77F1920
Сетевой адрес отправителя	194.85.135.65
Локальный (искомый) адрес получателя	008048EB7E60
Сетевой адрес получателя	194.85.135.75

В результате обмена ARP-сообщениями модуль IP, пославший запрос с интерфейса, имеющего адрес 194.85.135.75, определил, что IP-адресу 194.85.135.65 соответствует MAC-адрес 00E0F77F1920. Этот адрес затем помещается в заголовок кадра Ethernet, ожидавшего отправления IP-пакета.

Чтобы уменьшить число ARP-обращений в сети, найденное соответствие между IP-адресом и MAC-адресом сохраняется в ARP-таблице соответствующего интерфейса, в данном случае — это запись:

194.85.135.65 — 00E0F77F1920

Данная запись в ARP-таблице появляется автоматически, спустя несколько миллисекунд после того, как модуль ARP проанализирует ARP-ответ. Теперь, если вдруг вновь возникнет необходимость послать пакет по адресу 194.85.135.65, то протокол IP, прежде чем посылать широковещательный запрос, проверит, нет ли уже такого адреса в ARP-таблице.

ARP-таблица пополняется *не только за счет поступающих на данный интерфейс ARP-ответов*, но и в результате извлечения полезной информации из широковещательных ARP-запросов. Действительно, в каждом запросе, как это видно из табл. 14.3 и 14.4, содержатся IP- и MAC-адрес отправителя. Все интерфейсы, получившие этот запрос, могут поместить информацию о соответствии локального и сетевого адресов отправителя в собственную ARP-таблицу. В частности, все узлы, получившие ARP-запрос (см. табл. 14.3), могут пополнить свою ARP-таблицу записью:

194.85.135.75 — 008048EB7E60

Таким образом, вид ARP-таблицы, в которую в ходе работы сети были добавлены две упомянутые нами записи, иллюстрирует табл. 14.5.

Таблица 14.5. Пример ARP-таблицы

IP-адрес	MAC-адрес	Тип записи
194.85.135.65	00E0F77F1920	Динамический
194.85.135.75	008048EB7E60	Динамический
194.85.60.21	008048EB7567	Статический

В ARP-таблицах существуют два типа записей: динамические и статические. **Статические записи** создаются вручную с помощью утилиты arp и не имеют срока устаревания, точнее, они существуют до тех пор, пока компьютер или маршрутизатор остается включенным.

Динамические записи должны периодически обновляться. Если запись не обновлялась в течение определенного времени (порядка нескольких минут), то она исключается из таблицы. Таким образом, в ARP-таблице содержатся записи не обо всех узлах сети, а только о тех, которые активно участвуют в сетевых операциях. Поскольку такой способ хранения информации называют кэшированием, ARP-таблицы иногда называют **ARP-кэшем**.

ПРИМЕЧАНИЕ

Некоторые реализации протоколов IP и ARP не ставят IP-пакеты в очередь на время ожидания ARP-ответов. Вместо этого IP-пакет просто уничтожается, а его восстановление возлагается на модуль TCP или прикладной процесс, работающий через протокол UDP. Такое восстановление выполняется за счет тайм-аутов и повторных передач. Успешность повторной передачи обеспечивается первой попыткой, которая вызывает заполнение ARP-таблицы.

Совсем другой способ разрешения адресов используется *в глобальных сетях*, в которых не поддерживается широковещательная рассылка. Здесь администратору сети чаще всего приходится вручную формировать и помещать на какой-либо сервер ARP-таблицы, в которых он задает, например, соответствие IP-адресов адресам X.25, имеющих для протокола IP смысл локальных адресов. В то же время сегодня наметилась тенденция автоматизации работы протокола ARP и в глобальных сетях. Для этой цели среди всех маршрутизаторов, подключенных к какой-либо глобальной сети, выделяется специальный маршрутизатор, который ведет ARP-таблицу для всех остальных узлов и маршрутизаторов этой сети.

При таком централизованном подходе вручную нужно задать для всех узлов и маршрутизаторов только IP-адрес и локальный адрес выделенного для этих целей маршрутизатора. При включении каждый узел и маршрутизатор регистрируют свой адрес в выделенном маршрутизаторе. Всякий раз, когда возникает необходимость определения по IP-адресу локального адреса, модуль ARP обращается к выделенному маршрутизатору с запросом и автоматически получает ответ без участия администратора. Работающий таким образом маршрутизатор называют **ARP-сервером**.

В некоторых случаях возникает обратная задача — нахождение IP-адреса по известному локальному адресу. Тогда в действие вступает *реверсивный протокол разрешения адресов* (Reverse Address Resolution Protocol, RARP). Этот протокол используется, например, при старте бездисковых станций, не знающих в начальный момент времени своего IP-адреса, но знающих MAC-адрес своего сетевого адаптера.

Протокол Proxy-ARP

Протокол Proxy-ARP — это одна из разновидностей протокола ARP, позволяющая отображать IP-адреса на аппаратные адреса в сетях, поддерживающих широковещание, даже в тех случаях, когда искомый узел находится за пределами данного домена коллизий.

На рис. 14.9 показана сеть, один из конечных узлов которой (компьютер *D*) работает в **режиме удаленного узла**. В этом режиме узел обладает всеми возможностями компьютеров основной части сети Ethernet, в частности он имеет IP-адрес (IP_D), относящийся к той же сети. Для всех конечных узлов сети Ethernet особенности подключения удаленного узла (наличие модемов, коммутируемая связь, протокол PPP) абсолютно прозрачны — они взаимодействуют с ним обычным образом. Чтобы такой режим взаимодействия стал

возможным, среди прочего необходим протокол Proxy-ARP. Поскольку удаленный узел подключен к сети по протоколу PPP, то он, очевидно, *не имеет MAC-адреса*.

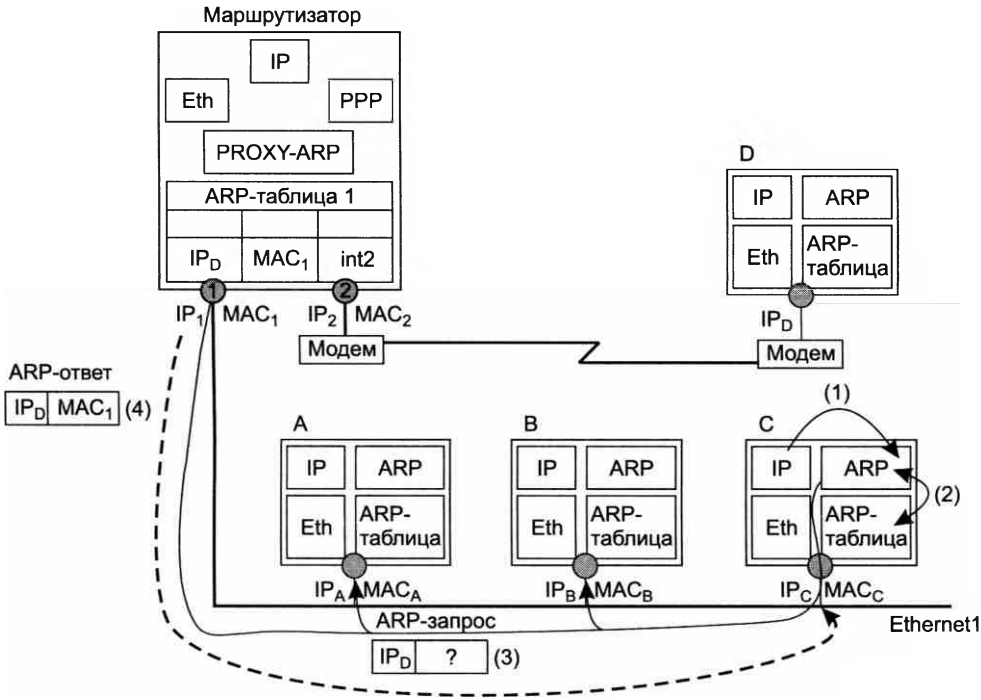


Рис. 14.9. Схема работы протокола Proxy-ARP

Пусть приложение, работающее, например, на компьютере *C*, решает послать пакет компьютеру *D*. Ему известен IP-адрес узла назначения (IP_D), однако, как мы уже не раз отмечали, для передачи пакета по сети Ethernet его необходимо упаковать в кадр Ethernet и снабдить MAC-адресом. Для определения MAC-адреса IP-протокол узла *C* обращается к протоколу ARP, который посылает широковещательное сообщение с ARP-запросом. Если бы в этой сети на маршрутизаторе не был установлен протокол Proxy-ARP, на этот запрос не откликнулся бы ни один узел.

Однако протокол Proxy-ARP установлен на маршрутизаторе и работает следующим образом. При подключении к сети удаленного узла *D* в таблицу ARP-маршрутизатора заносится запись

$$IP_D - MAC_1 - int2$$

Эта запись означает, что:

- при поступлении ARP-запроса на маршрутизатор относительно адреса IP_D в ARP-ответ будет помещен аппаратный адрес MAC_1, соответствующий аппаратному адресу интерфейса 1 маршрутизатора;
- узел, имеющий адрес IP_D, подключен к интерфейсу 2 маршрутизатора.

В ответ на посланный узлом *C* широковещательный ARP-запрос откликается маршрутизатор с установленным протоколом Proxy-ARP. Он посылает «ложный» ARP-ответ, в котором на место аппаратного адреса компьютера *D* помещает собственный адрес MAC_1. Узел *C*, не подозревая «подвоха», посылает кадр с IP-пакетом по адресу MAC_1. Получив кадр, маршрутизатор с установленным протоколом Proxy-ARP «понимает», что он направлен не ему (в пакете указан чужой IP-адрес) и поэтому надо искать адресата в ARP-таблице. Из таблицы видно, что кадр надо направить узлу, подключенному ко второму интерфейсу.

Мы рассмотрели простейшую схему применения протокола Proxy-ARP, которая, тем не менее, достаточно полно отражает логику его работы.

Система DNS

Пространство DNS-имен

В операционных системах, которые первоначально разрабатывались для локальных сетей, таких как Novell NetWare, Microsoft Windows или IBM OS/2, пользователи всегда работали с символьными именами компьютеров. Так как локальные сети состояли из небольшого числа компьютеров, применялись так называемые **плоские имена**, состоящие из последовательности символов, не разделенных на части. Примерами таких имен являются: NW1_1, mail2, MOSCOW_SALES_2. Для установления соответствия между символьными именами и MAC-адресами в этих операционных системах применялся механизм широковещательных запросов, подобный механизму запросов протокола ARP. Так, широковещательный способ разрешения имен реализован в протоколе NetBIOS, на котором были построены многие локальные ОС. Так называемые NetBIOS-имена стали на долгие годы одним из основных типов плоских имен в локальных сетях.

Для стека TCP/IP, рассчитанного в общем случае на работу в больших территориально распределенных сетях, подобный подход оказывается неэффективным.

В стеке TCP/IP применяется доменная система имен, которая имеет иерархическую древовидную структуру, допускающую наличие в имени произвольного количества составных частей (рис. 14.10).

Иерархия доменных имен аналогична иерархии имен файлов, принятой во многих популярных файловых системах. Дерево имен начинается с корня, обозначаемого здесь точкой. Затем следуют старшая символьная часть имени, вторая по старшинству символьная часть имени и т. д. Младшая часть имени соответствует конечному узлу сети. В отличие от имен файлов, при записи которых сначала указывается самая старшая составляющая, затем составляющая более низкого уровня и т. д., запись доменного имени начинается с самой младшей составляющей, а заканчивается самой старшей. Составные части доменного имени отделяются друг от друга точкой. Например, в имени home.microsoft.com составляющая home является именем одного из компьютеров в домене microsoft.com.

Разделение имени на части позволяет *разделить административную ответственность* за назначение уникальных имен между различными людьми или организациями в пределах своего уровня иерархии. Так, для приведенного примера один человек может нести ответственность за то, чтобы все имена с окончанием «ru» имели уникальную следующую вниз по иерархии часть. То есть все имена типа www.ru, mail.mmt.ru или m2.zil.mmt.ru отличаются второй по старшинству частью.

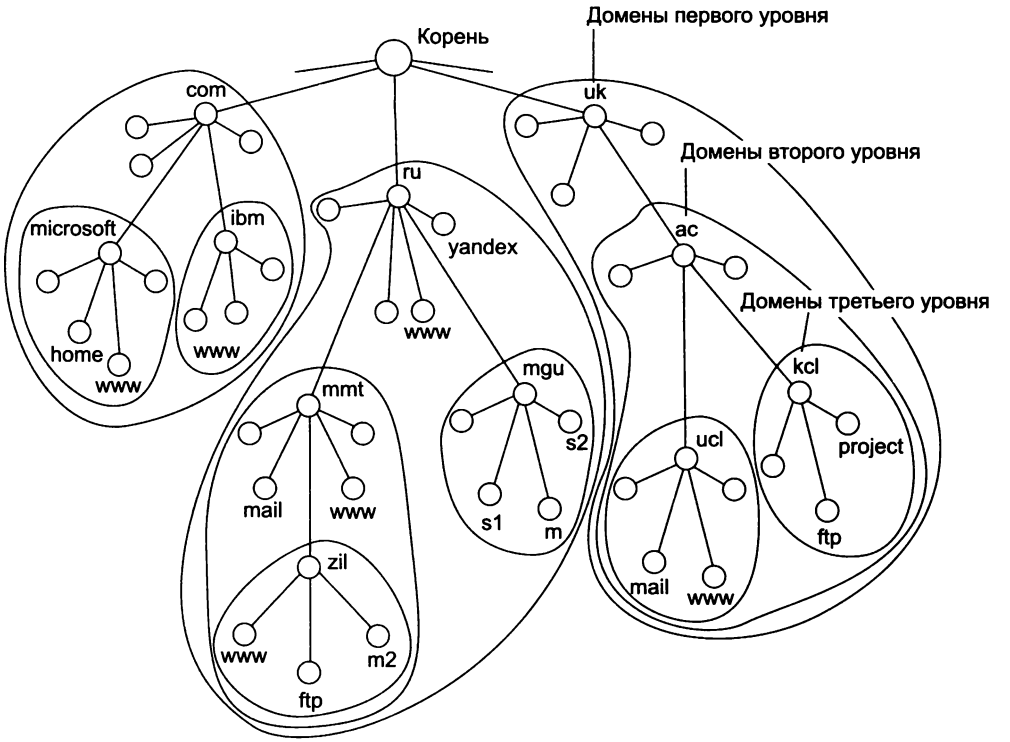


Рис. 14.10. Пространство доменных имен

Разделение административной ответственности позволяет решить проблему образования уникальных имен без взаимных консультаций между организациями, отвечающими за имена одного уровня иерархии. Очевидно, что должна существовать одна организация, отвечающая за назначение имен верхнего уровня иерархии.

Совокупность имен, у которых несколько старших составных частей совпадают, образует **домен имен** (domain). Например, имена www.zil.mmt.ru, ftp.zil.mmt.ru, yandex.ru и s1.mgu.ru входят в домен ru, так как все они имеют одну общую старшую часть — имя ru. Другим примером является домен mgu.ru. Из представленных на рис. 14.10 имен в него входят имена s1.mgu.ru, s2.mgu.ru и m.mgu.ru. Этот домен образуют имена, у которых две старшие части равны mgu.ru. Администратор домена mgu.ru несет ответственность за уникальность имен следующего уровня, входящих в домен, то есть имен s1, s2 и m. Образованные домены s1.mgu.ru, s2.mgu.ru и m.mgu.ru являются **поддоменами** домена mgu.ru, так как имеют общую старшую часть имени. Часто поддомены для краткости называют только младшей частью имени, то есть в нашем случае поддоменами являются s1, s2 и m.

О ТЕРМИНАХ

Термин «домен» очень многозначен, поэтому его нужно трактовать в рамках определенного контекста. Помимо доменов имен стека TCP/IP в компьютерной литературе часто упоминаются домены Windows NT, домены коллизий и некоторые другие. Общим у всех этих терминов является то, что они описывают некоторое множество компьютеров, обладающее каким-либо определенным свойством.

Если в каждом домене и поддомене обеспечивается уникальность имен следующего уровня иерархии, то и вся система имен будет состоять из уникальных имен.

По аналогии с файловой системой в доменной системе имен различают краткие, относительные и полные доменные имена. **Краткое доменное имя** — это имя конечного узла сети: хоста или порта маршрутизатора. Краткое имя — это лист дерева имен. **Относительное доменное имя** — это составное имя, начинающееся с некоторого уровня иерархии, но не самого верхнего. Например, www.zil — это относительное имя. **Полное доменное имя** включает составляющие всех уровней иерархии, начиная от краткого имени и кончая корневой точкой: www.zil.mmt.ru.

ВНИМАНИЕ

Компьютеры, имена которых относятся к одному и тому же домену, могут иметь абсолютно независимые друг от друга IP-адреса, принадлежащие разным сетям и подсетям. Например, в домен mgu.ru могут входить хосты с адресами 132.13.34.15, 201.22.100.33 и 14.0.0.6.

Корневой домен управляется центральными органами Интернета, в частности уже упоминавшейся нами организацией ICANN.

Домены верхнего уровня назначаются для каждой страны, а также для различных типов организаций. Имена этих доменов должны следовать международному стандарту ISO 3166. Для обозначения стран используются трехбуквенные и двухбуквенные аббревиатуры, например ru (Россия), uk (Великобритания), fi (Финляндия), us (Соединенные Штаты), а для различных типов организаций, например, следующие обозначения:

- com — коммерческие организации (например, microsoft.com);
- edu — образовательные организации (например, mit.edu);
- gov — правительственные организации (например, nsf.gov);
- org — некоммерческие организации (например, fidonet.org);
- net — сетевые организации (например, nsf.net).

Каждый домен администрирует отдельная организация, которая обычно разбивает свой домен на поддомены и передает функции администрирования этих поддоменов другим организациям.

Доменная система имен реализована в Интернете, но она может работать и в качестве автономной системы имен в любой крупной IP-сети, никак не связанной с Интернетом.

Иерархическая организация службы DNS

Широковещательный способ установления соответствия между символьными именами и локальными адресами, подобный реализованному в протоколе ARP, хорошо работает только в небольшой локальной сети, не разделенной на подсети. В крупных сетях, где возможность всеобщей широковещательной рассылки не поддерживается, нужен другой способ разрешения символьных имен. Альтернативой широковещательной рассылке является применение *централизованной службы*, поддерживающей соответствие между символьными именами и IP-адресами всех компьютеров сети.

На раннем этапе развития Интернета на каждом хосте вручную создавался текстовый **файл отображений** с известным именем hosts.txt. Этот файл состоял из некоторого ко-

личества строк, каждая из которых содержала одну пару «доменное имя — IP-адрес», например:

rhino.acme.com — 102.54.94.97

По мере роста Интернета файлы hosts.txt также увеличивались в объеме и создание *масштабируемого* решения для разрешения имен стало необходимостью. Таким решением стала централизованная служба **DNS** (Domain Name System — система доменных имен), основанная на распределенной базе отображений «доменное имя — IP-адрес».

Служба DNS имеет иерархическую структуру. Иерархию образуют **DNS-серверы**, которые поддерживают распределенную базу отображений, а **DNS-клиенты** обращаются к серверам с запросами об отображении доменного имени на IP-адрес (говорят также «о разрешении» доменного имени). DNS-клиентом является практически каждый узел Интернета, будь то клиентский компьютер, сервер приложений или маршрутизатор.

ПРИМЕЧАНИЕ

Клиентом службы DNS является программное обеспечение ОС, называемое резольвером. Приложения ОС сами не обращаются напрямую к службе DNS, а обращаются к резольверу своей ОС. Резольверу должен быть известен IP-адрес по крайней мере одного DNS-сервера. Подавляющее большинство DNS-серверов использует программное обеспечение BIND (Berkeley Internet Name Domain), первоначально разработанное в Калифорнийском университете Беркли.

Запросы к DNS-серверам и их ответы обслуживаются **протоколом DNS**, что позволяет клиенту делать запросы относительно некоторого доменного имени, либо задавая тип записи, либо запрашивая все типы, относящиеся к данному имени. DNS-сообщения чаще всего передаются в дейтаграммах UDP с портом сервера 53, но в некоторых случаях, требующих повышенной надежности, служба DNS обращается к услугам TCP.

Служба DNS использует текстовые файлы почти такого же формата, как и файл hosts, состоящие из записей отображений и некоторых служебных записей. Эти файлы также подготавливаются вручную администраторами сетей. Однако поскольку служба DNS опирается на иерархию доменов, где для каждого домена создается свой DNS-сервер, ее эффективность и масштабируемость несравнимы с ручной процедурой разрешения имен. При росте количества узлов в сети проблема масштабирования решается созданием новых доменов и поддоменов имен и добавлением в службу DNS новых серверов.

Вершину иерархии серверов DNS составляют **корневые серверы**, они хранят файлы отображений DNS-серверов следующего уровня, называемого *верхним* (top level DNS). Серверы верхнего уровня хранят данные об именах и адресах имен, входящих в домены верхнего уровня, таких как com, ru или fm, а также об именах DNS-серверов, которые обслуживают домены следующего уровня иерархии — второго, такие как cisco.com или yandex.ru.

Разделение пространства имен между серверами

DNS-сервер отвечает на запросы клиентов на основе информации, содержащейся в текстовых файлах отображений имен, хранящихся на данном сервере. В принципе, DNS-сервер мог бы хранить данные всех отображений, входящих в некоторый домен со всеми его поддоменами; при таком подходе сервер верхнего уровня, отвечающий, например, за домен com, хранил бы в своих файлах записи всех имен, заканчивающихся на com: ibm.com,

www.ibm.com, www2.ibm.com, cisco.com, www.cisco.com и т. д. Понятно, что такой подход не масштабируем и не может работать в Интернете.

Поэтому пространство доменных имен «разрезают» между DNS-серверами обычно так, чтобы сервер хранил записи только в пределах одного уровня, а для имен своих поддоменов хранил только ссылки на DNS-серверы, отвечающие за эти поддомены. Например, DNS-сервер верхнего уровня, отвечающий за домен com, хранит только записи о листьях своего домена, например имя www.com, а также об именах DNS-серверов, которые обслуживают поддомены домена com, например DNS-сервера поддомена cisco.com.

Часть пространства доменных имен, для которых некоторый DNS-сервер имеет полную информацию об их отображениях на основе соответствующего текстового файла, называется **зоной DNS**, а сам текстовый файл — **файлом зоны**. Когда DNS-сервер дает ответ о записи, входящей в зону, за которую он отвечает, такой ответ называется *полномочным* (authoritative) *ответом DNS*. Как мы увидим далее, DNS-сервер может также давать *неполномочный* ответ, если запрос относится не к его зоне, но он знает его за счет кэширования ответов других серверов. Заметим, что DNS-сервер может обслуживать несколько зон.

Файл зоны состоит из текстовых записей нескольких типов, таких как:

- A — отображает имя на IPv4-адрес;
- AAAA — отображает имя на IPv6-адрес;
- NS — определяет имя DNS-сервера для некоторого домена;
- MS — определяет имя почтового сервера для некоторого домена.

Существуют записи и некоторых других типов.

Для обеспечения надежности и высокой производительности для каждой зоны существуют один *первичный* и несколько *вторичных* DNS-серверов. На первичном сервере находится *мастер-копия* файла зоны, которая редактируется администратором сервера. Вторичные серверы периодически копируют файл зоны с первичного сервера, для этого может использоваться протокол DNS, в котором имеется соответствующий тип запроса, или же администратор может задействовать любой протокол копирования файлов, например ftp или scp. В том случае, когда файл зоны передается по протоколу DNS, для повышения надежности применяется протокол TCP (порт 53).

Рекурсивная и нерекурсивная процедуры

Существуют две основные схемы разрешения DNS-имен. В первом варианте работу по поиску IP-адреса координирует DNS-клиент:

1. DNS-клиент обращается к корневому DNS-серверу с указанием полного доменного имени.
2. DNS-сервер отвечает клиенту, указывая адрес следующего DNS-сервера, обслуживающего домен верхнего уровня, заданный в следующей старшей части запрошенного имени.
3. DNS-клиент делает запрос следующего DNS-сервера, который отсылает его к DNS-серверу нужного поддомена, и т. д., пока не будет найден DNS-сервер, в котором хранится отображение запрошенного имени на IP-адрес. Этот сервер дает окончательный ответ клиенту.

Такая процедура разрешения имени называется *нерекурсивной* — в этом случае клиент сам итеративно выполняет последовательность запросов к разным серверам имен. Данная схема переносит бóльшую часть работы по разрешению имени на клиента, и она применяется редко.

Во втором варианте реализуется *рекурсивная* процедура:

1. DNS-клиент запрашивает локальный DNS-сервер, то есть тот сервер, обслуживающий поддомен, которому принадлежит имя клиента.
2. Далее возможны два варианта действий:
   - если локальный DNS-сервер знает ответ, то он сразу же возвращает его клиенту (это может произойти, когда запрошенное имя входит в тот же поддомен, что и имя клиента, или когда сервер уже узнавал данное соответствие для другого клиента и сохранил его в своем кэше);
   - если локальный сервер не знает ответ, то он выполняет итеративные запросы к корневому серверу и т. д. точно так же, как это делал клиент в предыдущем варианте, а получив ответ, передает его клиенту, который все это время просто ждет его от своего локального DNS-сервера.

В этой схеме клиент перепоручает работу своему серверу, именно поэтому схема называется рекурсивной, или косвенной. Практически все резольверы используют или по крайней мере запрашивают в качестве приоритетной рекурсивную процедуру.

DNS-серверы стараются не поддерживать рекурсивный режим ответов, так как это перегружает их; корневые серверы и серверы верхнего уровня всегда дают нерекурсивные ответы, отсылая серверы нижних уровней к серверам промежуточных уровней.

Получая окончательный ответ от сервера вышестоящего уровня, рекурсивный сервер кэширует его для того, чтобы при поступлении аналогичного запроса дать быстрый неполномочный ответ. Чтобы служба DNS могла оперативно отрабатывать изменения, происходящие в сети, ответы кэшируются на относительно короткое время — обычно от нескольких часов до нескольких дней, срок жизни записи задается администратором полномочного сервера.

DNS-сервер может быть *открытым* — в этом случае он отвечает любому клиенту, или же *закрытым* — в этом случае он отвечает либо только клиентам своего предприятия (в случае корпоративного сервера), либо только своим подписчикам услуг доступа в Интернет (в случае провайдера).

Корневые серверы

Корневые серверы — наиболее уязвимое звено службы DNS, так как разрешение всех запросов, ответы на которые не находятся в кэше или файле зоны какого-либо DNS-сервера нижнего уровня, начинаются с обращения к одному из корневых серверов. Разработчики системы DNS (в начале 80-х годов) понимали это, поэтому уже изначально было решено обеспечить высокую степень резервирования: было установлено 13 корневых серверов с именами от a.root-servers.net, b.root-servers.net, c.root-servers.net... m.root-servers.net и тринадцатью IP-адресами.

С тех пор организация корневых DNS-серверов изменилась, вместо 13 серверов Интернет обслуживает более 300 — в августе 2013 года их было 376 (их географическое распреде-

ление показано на рис. 14.11). Это значительно повысило отказоустойчивость и производительность службы DNS. Все корневые серверы по-прежнему разделяют те же 13 имен (от a.root-servers.org до m.root-servers.org) и 13 IP-адресов. Только теперь каждому имени и адресу соответствует кластер серверов. Например, имени f.root-servers.net соответствует 56 серверов, а имени l.root-servers.net — 146. Корневые серверы распределены географически, а каждый кластер, соответствующий одному имени, администрируется отдельной организацией.

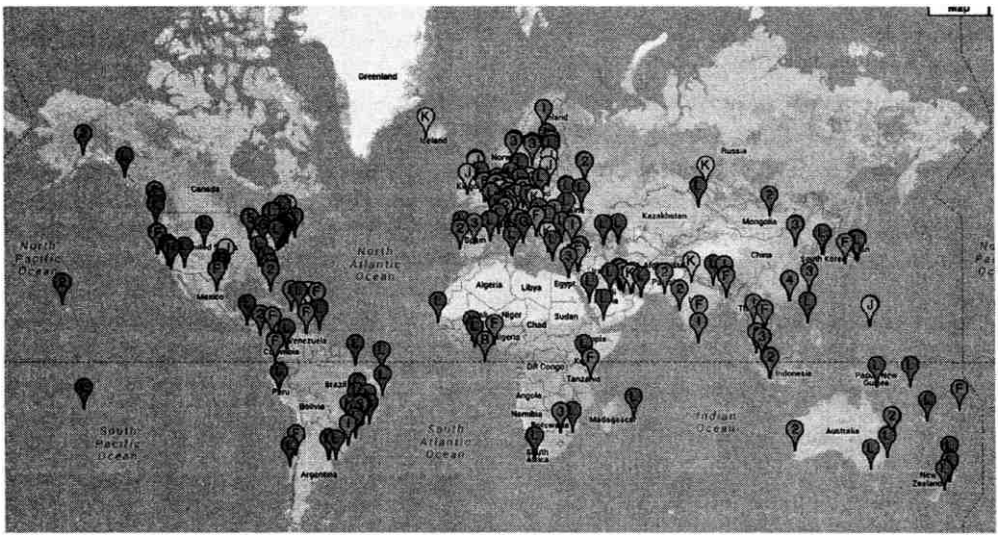


Рис. 14.11. Географическое распределение корневых серверов DNS (источник: rooot-servers.org)

Использование произвольной рассылки

Служба DNS является одним из примеров успешного использования для адресации и маршрутизации техники **произвольной рассылки** (anycast)[1].

ПРИМЕЧАНИЕ

Техника произвольной рассылки основана на использовании обычных IP-адресов (unicast) и стандартных механизмов маршрутизации IP-сетей. Не существует признака (такого, например, как определенное значение нескольких старших разрядов), по которому IP-адрес можно было бы отнести к классу адресов произвольной рассылки. Техника произвольной рассылки является надстройкой над имеющимися стандартными транспортными средствами IP-сетей. При этом разработчики техники произвольной рассылки учли и исключили возможность возникновения коллизий, причинами которых может явиться дублирование IP-адресов.

В DNS-службе техника произвольной рассылки используется для рационализации взаимодействия клиента и серверов. Пусть имеется некоторая группа DNS-серверов, предоставляющих клиентам идентичные услуги. Клиенту не важно, к какому из узлов данной группы будет

[1] Об адресах произвольной рассылки читайте в главах 2 и 15.

передан его запрос. В соответствии с технологией произвольной рассылки всем серверам группы должен быть присвоен один и тот же IP-адрес, который в данной ситуации интерпретируется как адрес произвольной рассылки. Кроме того, должны быть найдены маршруты от DNS-клиента до каждого из серверов группы. При отправке запроса к серверам группы клиент выбирает в соответствии с некоторыми правилами предпочтения один из маршрутов (серверов). В случае службы DNS клиент обычно выбирает ближайший сервер.

Использование в службе DNS техники произвольной рассылки сулит несколько потенциальных преимуществ:

- повышение производительности за счет распараллеливания нагрузки на серверы (баланс нагрузки);
- повышение надежности за счет «горячего» резервирования серверов, когда любой сервер может выполнить запрос клиента;
- защита от DDoS/DoS-атак[1] — чтобы вывести из строя все серверы, атакующему придется проводить одновременную атаку на большое число серверов и сетей, что затруднительно даже для большой армии ботов.

Обратная зона

Служба DNS предназначена не только для нахождения IP-адреса по имени хоста, но и для решения *обратной задачи* — нахождения DNS-имени по известному IP-адресу.

Многие программы и утилиты, пользующиеся службой DNS, пытаются найти имя узла по его адресу в том случае, когда пользователем задан только адрес (или этот адрес программа узнала из пришедшего пакета). Обратная запись не всегда существует даже для тех адресов, для которых есть прямые записи. Ее могут просто забыть создать или же ее создание требует дополнительной оплаты. Обратная задача решается в Интернете путем организации так называемых обратных зон.

Обратная зона — это система таблиц, которая хранит соответствие между IP-адресами и DNS-имена хостов некоторой сети. Для организации распределенной службы и использования для поиска имен того же программного обеспечения, что и для поиска адресов, применяется оригинальный подход, связанный с представлением IP-адреса в виде DNS-имени.

Первый этап преобразования заключается в том, что составляющие IP-адреса интерпретируются как составляющие DNS-имени. Например, адрес 192.31.106.0 рассматривается как состоящий из старшей части, соответствующей домену 192, затем идет домен 31, в который входит домен 106.

Далее, учитывая, что при записи IP-адреса старшая часть является самой *левой* частью адреса, а при записи DNS-имени — самой *правой*, то составляющие в преобразованном адресе указываются в обратном порядке, то есть для данного примера — 106.31.192.

Для хранения отображений всех адресов, начинающихся, например, с числа 192, заводится зона 192 со своими серверами имен. Для записей о серверах, поддерживающих старшие в иерархии обратные зоны, создана специальная зона in-addr.arpa, поэтому полная запись для использованного в примере адреса выглядит так:

```
106.31.192.in-addr.arpa.
```

[1] Читайте об атаках на DNS-серверы в главе 29.

Серверы для обратных зон используют файлы баз данных, не зависящие от файлов основных зон, в которых имеются записи о прямом соответствии тех же имен и адресов. Такая организация данных может приводить к несогласованности, так как одно и то же соответствие вводится в файлы дважды.

Протокол DHCP

Для нормальной работы сети каждому сетевому интерфейсу компьютера и маршрутизатора должен быть назначен IP-адрес.

Процедура присвоения адресов происходит в ходе **конфигурирования** компьютеров и маршрутизаторов. Назначение IP-адресов может происходить вручную в результате выполнения процедуры конфигурирования интерфейса, для компьютера сводящейся, например, к заполнению системы экранных форм. При этом администратор должен помнить, какие адреса из имеющегося множества он уже использовал для других интерфейсов, а какие еще свободны. При конфигурировании помимо IP-адресов сетевых интерфейсов (и соответствующих масок) устройству сообщается ряд других **конфигурационных параметров**. При конфигурировании администратор должен назначить клиенту не только IP-адрес, но и другие параметры стека TCP/IP, необходимые для его эффективной работы, например маску и IP-адрес маршрутизатора, предлагаемые по умолчанию, IP-адрес DNS-сервера, доменное имя компьютера и т. п. Даже при не очень большом размере сети эта работа представляет для администратора утомительную процедуру.

Протокол динамического конфигурирования хостов (Dynamic Host Configuration Protocol, DHCP) автоматизирует процесс конфигурирования сетевых интерфейсов, гарантируя от дублирования адресов за счет централизованного управления их распределением.

Режимы DHCP

Протокол DHCP работает в соответствии с моделью *клиент-сервер*. Во время старта системы компьютер, являющийся DHCP-клиентом, посылает в сеть широковещательный запрос на получение IP-адреса. DHCP-сервер откликается и посылает сообщение-ответ, содержащее IP-адрес и некоторые другие конфигурационные параметры.

При этом DHCP-сервер может работать в разных режимах, включая:

- ручное назначение статических адресов;
- автоматическое назначение статических адресов;
- автоматическое распределение динамических адресов.

Во всех режимах работы администратор при конфигурировании DHCP-сервера сообщает ему один или несколько диапазонов IP-адресов, причем все эти адреса относятся к одной сети, то есть имеют одно и то же значение в поле номера сети.

В **ручном** режиме администратор помимо пула доступных адресов снабжает DHCP-сервер информацией о жестком соответствии IP-адресов физическим адресам или другим идентификаторам клиентских узлов. DHCP-сервер, пользуясь этой информацией, *всегда* выдаст

определенному DHCP-клиенту *один и тот же* назначенный ему администратором IP-адрес (а также набор других конфигурационных параметров[1]).

В режиме **автоматического** назначения статических адресов DHCP-сервер самостоятельно, без вмешательства администратора, произвольным образом выбирает клиенту IP-адрес из пула наличных IP-адресов. Адрес дается клиенту из пула в постоянное пользование, то есть между идентифицирующей информацией клиента и его IP-адресом по-прежнему, как и при ручном назначении, существует постоянное соответствие. Оно устанавливается в момент первого назначения DHCP-сервером IP-адреса клиенту. При всех последующих запросах сервер возвращает клиенту тот же самый IP-адрес.

При **динамическом** распределении адресов DHCP-сервер выдает адрес клиенту на ограниченное время, называемое **сроком аренды**. Когда компьютер, являющийся DHCP-клиентом, удаляется из подсети, назначенный ему IP-адрес автоматически освобождается. Когда компьютер подключается к другой подсети, то ему автоматически назначается новый адрес. Ни пользователь, ни сетевой администратор не вмешиваются в этот процесс.

Это дает возможность впоследствии повторно использовать этот IP-адрес для назначения другому компьютеру. Таким образом, помимо основного преимущества DHCP — автоматизации рутинной работы администратора по конфигурированию стека TCP/IP на каждом компьютере, режим динамического распределения адресов в принципе позволяет строить IP-сеть, количество узлов в которой превышает количество имеющихся в распоряжении администратора IP-адресов.

Покажем преимущества, которые дает динамическое распределение пула адресов, на примере. Пусть в некоторой организации сотрудники значительную часть рабочего времени проводят вне офиса — дома или в командировках. Каждый из них имеет портативный компьютер, который во время пребывания в офисе подключается к корпоративной IP-сети. Возникает вопрос, сколько IP-адресов необходимо этой организации?

Первый ответ — столько, *скольким сотрудникам необходим доступ в сеть*. Если их 500 человек, то каждому из них должен быть назначен IP-адрес и выделено рабочее место. То есть администрация должна получить у поставщика услуг адреса двух сетей класса C и оборудовать соответствующим образом помещение. Однако вспомним, что сотрудники в этой организации редко появляются в офисе, значит, бóльшая часть ресурсов при таком решении будет простаивать.

Второй ответ — столько, *сколько сотрудников обычно присутствует в офисе* (с некоторым запасом). Если обычно в офисе работает не более 50 сотрудников, то достаточно получить у поставщика услуг пул из 64 адресов и установить в рабочем помещении сеть с 64 коннекторами для подключения компьютеров. Но возникает другая проблема — кто и как будет конфигурировать компьютеры, состав которых постоянно меняется?

Существуют два пути. Во-первых, администратор (или сам мобильный пользователь) может конфигурировать компьютер вручную каждый раз, когда возникает необходимость подключения к офисной сети. Такой подход требует от администратора (или пользователей) большого объема рутинной работы, следовательно — это плохое решение. Гораздо привлекательнее выглядят возможности автоматического динамического назначения DHCP-адресов. Действительно, администратору достаточно один раз при настройке DHCP-сервера указать диапазон из 64 адресов, а каждый вновь прибывающий мобильный

[1] Далее для краткости это уточнение будет опускаться.

пользователь будет просто физически подключать в сеть свой компьютер, на котором запускается DHCP-клиент. Он запросит конфигурационные параметры и автоматически получит их от DHCP-сервера. Таким образом, для работы 500 мобильных сотрудников достаточно иметь в офисной сети 64 IP-адреса и 64 рабочих места.

Алгоритм динамического назначения адресов

Администратор управляет процессом конфигурирования сети, определяя два основных конфигурационных параметра DHCP-сервера: *пул адресов, доступных распределению*, и *срок аренды*. Срок аренды диктует, как долго компьютер может использовать назначенный IP-адрес, перед тем как снова запросить его у DHCP-сервера. Срок аренды зависит от режима работы пользователей сети. Если это небольшая сеть учебного заведения, куда со своими компьютерами приходят многочисленные студенты для выполнения лабораторных работ, то срок аренды может быть равен длительности лабораторной работы. Если же это корпоративная сеть, в которой сотрудники предприятия работают на регулярной основе, то срок аренды может быть достаточно длительным — несколько дней или даже недель.

DHCP-сервер должен находиться в одной подсети с клиентами, учитывая, что клиенты посылают ему широковещательные запросы (рис. 14.12). Для снижения риска выхода сети из строя из-за отказа DHCP-сервера в сети иногда ставят резервный DHCP-сервер (такой вариант соответствует сети 1).

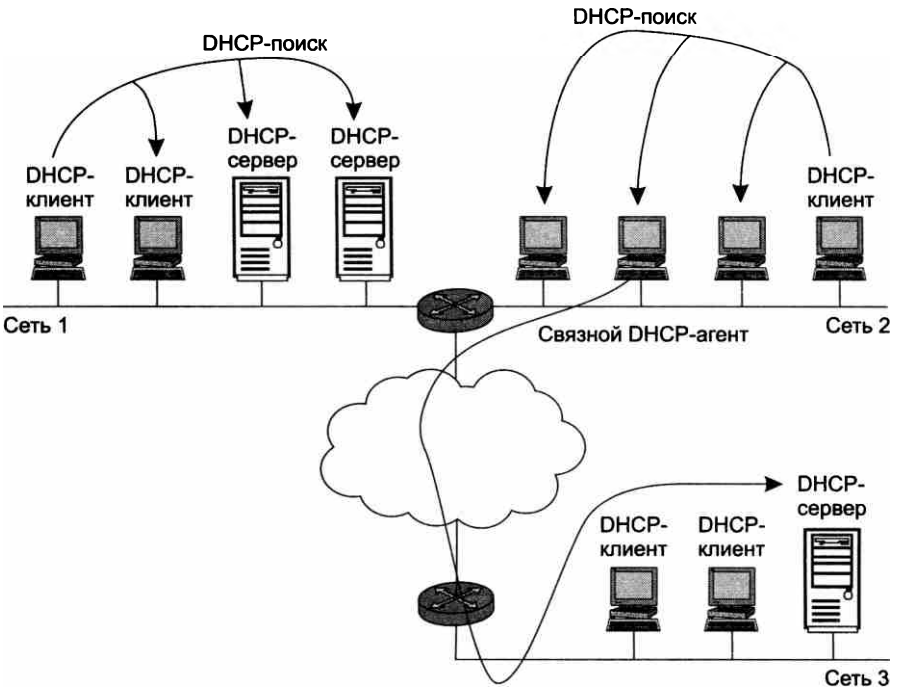


Рис. 14.12. Схемы взаимного расположения DHCP-серверов и DHCP-клиентов

Иногда наблюдается и обратная картина: в сети нет ни одного DHCP-сервера. В этом случае его подменяет связной **DHCP-агент** — программное обеспечение, играющее роль посредника между DHCP-клиентами и DHCP-серверами (пример такого варианта — сеть 2). Связной агент переправляет запросы клиентов из сети 2 DHCP-серверу сети 3. Таким образом, один DHCP-сервер может обслуживать DHCP-клиентов нескольких разных сетей.

Вот как выглядит упрощенная схема обмена сообщениями между клиентскими и серверными частями DHCP.

1. Когда компьютер включают, установленный на нем DHCP-клиент посылает ограниченное широковещательное сообщение DHCP-поиска (IP-пакет с адресом назначения, состоящим из одних единиц, который должен быть доставлен всем узлам данной IP-сети).
2. Находящиеся в сети DHCP-серверы получают это сообщение. Если в сети DHCP-серверы отсутствуют, то сообщение DHCP-поиска получает связной DHCP-агент. Он пересылает это сообщение в другую, возможно, значительно отстоящую от него сеть DHCP-серверу, IP-адрес которого ему заранее известен.
3. Все DHCP-серверы, получившие сообщение DHCP-поиска, посылают DHCP-клиенту, обратившемуся с запросом, свои DHCP-предложения. Каждое предложение содержит IP-адрес и другую конфигурационную информацию. (DHCP-сервер, находящийся в другой сети, посылает ответ через агента.)
4. DHCP-клиент собирает конфигурационные DHCP-предложения от всех DHCP-серверов. Как правило, он выбирает первое из поступивших предложений и отправляет в сеть широковещательный DHCP-запрос. В этом запросе содержатся идентификационная информация о DHCP-сервере, предложение которого принято, а также значения принятых конфигурационных параметров.
5. Все DHCP-серверы получают DHCP-запрос, и только один выбранный DHCP-сервер посылает положительную DHCP-квитанцию (подтверждение IP-адреса и параметров аренды), а остальные серверы аннулируют свои предложения, в частности возвращают в свои пулы предложенные адреса.
6. DHCP-клиент получает положительную DHCP-квитанцию и переходит в рабочее состояние.

Время от времени компьютер пытается обновить параметры аренды у DHCP-сервера. Первую попытку он делает задолго до истечения срока аренды, обращаясь к тому серверу, от которого он получил текущие параметры. Если ответа нет или ответ отрицательный, он через некоторое время снова посылает запрос. Так повторяется несколько раз, и если все попытки получить параметры у того же сервера оказываются безуспешными, клиент обращается к другому серверу. Если и другой сервер отвечает отказом, то клиент теряет свои конфигурационные параметры и переходит в режим автономной работы.

Также DHCP-клиент может по своей инициативе досрочно отказаться от выделенных ему параметров.

В сети, где адреса назначаются динамически, нельзя быть уверенным в адресе, который в данный момент имеет тот или иной узел. И такое непостоянство IP-адресов влечет за собой некоторые проблемы.

Во-первых, *возникают сложности при преобразовании символьного доменного имени в IP-адрес*. Действительно, представьте себе функционирование системы DNS, которая должна поддерживать таблицы соответствия символьных имен IP-адресам в условиях, когда последние меняются каждые два часа! Учитывая это обстоятельство, для серверов, к которым пользователи часто обращаются по символьному имени, назначают статические IP-адреса, оставляя динамические только для клиентских компьютеров. Однако в некоторых сетях количество серверов настолько велико, что их ручное конфигурирование становится слишком обременительным. Это привело к разработке усовершенствованной версии DNS (так называемой динамической системы DNS), в основе которой лежит согласование информационной адресной базы в службах DHCP и DNS.

Во-вторых, *трудно осуществлять удаленное управление и автоматический мониторинг интерфейса* (например, сбор статистики), если в качестве его идентификатора выступает динамически изменяемый IP-адрес.

Наконец, для обеспечения безопасности сети многие сетевые устройства могут блокировать (фильтровать) пакеты, определенные поля которых имеют некоторые заранее заданные значения. Другими словами, при динамическом назначении адресов *усложняется фильтрация пакетов по IP-адресам*.

Последние две проблемы проще всего решаются отказом от динамического назначения адресов для интерфейсов, фигурирующих в системах мониторинга и безопасности.

Выводы

В стеке TCP/IP используются три типа адресов: локальные (называемые также аппаратными), IP-адреса и символьные доменные имена. Все эти типы адресов присваиваются узлам составной сети независимо друг от друга.

IP-адрес имеет длину 4 байта и состоит из номера сети и номера узла. Для определения границы, отделяющей номер сети от номера узла, сегодня используются два подхода. Первый основан на применении классов адресов, второй — масок.

Номер сети назначается централизованно, если сеть является частью Интернета. Назначение IP-адресов узлам сети может происходить либо вручную (администратор сам ведет списки свободных и занятых адресов и конфигурирует сетевой интерфейс), либо автоматически (с использованием протокола DHCP).

Установление соответствия между IP-адресом и аппаратным адресом сетевого интерфейса осуществляется протоколом разрешения адресов (ARP).

В стеке TCP/IP применяется система доменных символьных имен, которая имеет иерархическую древовидную структуру, допускающую использование в имени произвольного количества составных частей. Совокупность имен, у которых несколько старших составных частей совпадают, образует домен имен. Доменные имена назначаются централизованно, если сеть является частью Интернета, в противном случае — локально.

Соответствие между доменными именами и IP-адресами может устанавливаться как средствами локального хоста с использованием файла hosts, так и с помощью централизованной службы DNS.

Контрольные вопросы

1. Какие из приведенных адресов не могут быть использованы в качестве IP-адресов сетевого интерфейса для узлов Интернета? Для синтаксически правильных адресов определите их класс: A, B, C, D или E. Варианты адресов:

   а) 223.13.123.245;

   б) 225.0.0.105;

   в) 194.87.45.0;

   г) 10.24.255.252;

   д) 125.24.255.255;

   е) 157.213.255.305;

   ж) 129.12.255.255;

   з) 127.0.23.255;

   и) 1.0.0.13;

   к) 221.1.1.1;

   л) 192.134.216.255;

   м) 193.256.254.11.

2. Пусть IP-адрес некоторого узла подсети равен 108.5.18.167, а значение маски для этой подсети — 255.255.240.0. Определите номер подсети. Какое максимальное число сетевых интерфейсов может быть в этой подсети?

3. Пусть вам ничего не известно о структуре сети, но в вашем распоряжении имеется следующая таблица соответствия IP-адресов и DNS-имен нескольких узлов сети:

IP-адрес узла	123.1.0.01	123.1.0.02	123.1.0.03	123.1.0.04	?	?
DNS-имя узла	w1.mgu.ru	w2.mgu.ru	w3.mgu.ru	w4.mgu.ru	w5.mgu.ru	w6.mgu.ru

   Что вы можете сказать об IP-адресах узлов, имеющих DNS-имена w5.mgu.ru и w6.mgu.ru?

4. Пусть вам ничего не известно о структуре сети, но вы знаете DNS-имена некоторых узлов: w1.mgu.ru, w4.mgu.ru и w3.dept.ru. Что вы можете сказать о том, насколько близко территориально находятся они относительно друг друга? Варианты ответов:

   а) узел w1.mgu.ru расположен ближе к w6.mgu.ru, чем к w3.dept.ru;

   б) узел w1.mgu.ru расположен ближе к w3.dept.ru, чем к w6.mgu.ru;

   в) ничего определенного.

5. Какое максимальное количество подсетей теоретически можно организовать, если в вашем распоряжении имеется сеть класса B? Какое значение должна при этом иметь маска?

6. Протокол ARP функционально можно разделить на клиентскую и серверную части. Опишите, какие функции вы отнесли бы к клиентской части, а какие — к серверной?

ГЛАВА 15 Протокол межсетевого взаимодействия

IP-пакет

В каждой очередной сети, лежащей на пути перемещения пакета, протокол IP обращается к средствам транспортировки этой сети, чтобы с их помощью передать пакет на маршрутизатор, ведущий к следующей сети, или непосредственно на узел-получатель. *Поддержание интерфейса с нижележащими технологиями* подсетей является одной из важнейших функций протокола IP. В эти функции входит также *поддержание интерфейса с протоколами вышележащего транспортного уровня*, в частности с протоколом TCP, который решает все вопросы обеспечения надежной доставки данных по составной сети в стеке TCP/IP.

Протокол IP относится к протоколам *без установления соединений*, он поддерживает обработку каждого IP-пакета как независимой единицы обмена, не связанной с другими пакетами. В протоколе IP нет механизмов, обычно применяемых для обеспечения достоверности конечных данных. Если во время продвижения пакета происходит какая-либо ошибка, то протокол IP по своей инициативе ничего не предпринимает для исправления этой ошибки. Например, если на промежуточном маршрутизаторе пакет был отброшен из-за ошибки по контрольной сумме, то модуль IP не пытается заново послать потерянный пакет. Другими словами, протокол IP реализует политику доставки «по возможности».

Имеется прямая связь между количеством полей заголовка пакета и функциональной сложностью протокола, который работает с этим заголовком. Чем проще заголовок — тем проще соответствующий протокол. Бóльшая часть действий протокола связана с обработкой той служебной информации, которая переносится в полях заголовка пакета. Изучая назначение каждого поля заголовка IP-пакета, мы не только получаем формальные знания о структуре пакета, но и знакомимся с основными функциями протокола IP. На рис. 15.1 показаны поля заголовка IP-пакета.

Поле **номера версии** занимает 4 бита и идентифицирует версию протокола IP. Сейчас повсеместно используется версия 4 (IPv4), хотя все чаще встречается и новая версия (IPv6).

Значение **длины заголовка** IP-пакета также занимает 4 бита и измеряется в 32-битных словах. Обычно заголовок имеет длину в 20 байт (пять 32-битных слов), но при добавлении некоторой служебной информации это значение может быть увеличено за счет дополнительных байтов в поле параметров. Наибольшая длина заголовка составляет 60 байт.

Поле **типа сервиса** (Type of Service, ToS) имеет и другое, более современное название — **байт дифференцированного обслуживания**, или **DS-байт**. Этим двум названиям соответствуют два варианта интерпретации этого поля. В обоих случаях данное поле служит одной цели — хранению признаков, которые отражают требования к качеству обслуживания пакета. В прежнем варианте первые три бита содержат значение **приоритета** пакета: от самого низкого — 0 до самого высокого — 7. Маршрутизаторы и компьютеры могут

Рис. 15.1. Структура заголовка IP-пакета

принимать во внимание приоритет пакета и обрабатывать более важные пакеты в первую очередь. Следующие три бита поля ToS определяют **критерий выбора маршрута**. Если бит D (Delay — задержка) установлен в 1, то маршрут должен выбираться для минимизации задержки доставки данного пакета, установленный бит T (Throughput — пропускная способность) — для максимизации пропускной способности, а бит R (Reliability — надежность) — для максимизации надежности доставки. Оставшиеся два бита имеют нулевое значение.

Стандарты дифференцированного обслуживания, принятые в конце 90-х годов, дали новое название этому полю и переопределили назначение его битов. В DS-байте также используются только старшие 6 бит, а два младших бита остаются в качестве резерва. Назначение битов DS-байта рассмотрено в разделе «Поддержка QoS в машрутизаторах» главы 17.

Поле **общей длины** занимает 2 байта и характеризует общую длину пакета с учетом заголовка и поля данных. Максимальная длина пакета ограничена разрядностью поля, определяющего эту величину, и составляет 65 535 байт, однако в большинстве компьютеров и сетей столь большие пакеты не используются. При передаче по сетям различного типа длина пакета выбирается с учетом максимальной длины пакета протокола нижнего уровня, несущего IP-пакеты. Если это кадры Ethernet, то выбираются пакеты с максимальной длиной 1500 байт, умещающиеся в поле данных кадра Ethernet. В стандартах TCP/IP предусматривается, что все хосты должны быть готовы принимать пакеты длиной вплоть до 576 байт (независимо от того, приходят ли они целиком или фрагментами).

Идентификатор пакета занимает 2 байта и используется для распознавания пакетов, образовавшихся путем деления на части (фрагментации) исходного пакета. Все части (фрагменты) одного пакета должны иметь одинаковое значение этого поля.

Флаги занимают 3 бита и содержат признаки, связанные с фрагментацией. Установленный в 1 бит DF (Do not Fragment — не фрагментировать) запрещает маршрутизатору фрагментировать данный пакет, а установленный в 1 бит MF (More Fragments — больше фрагментов) говорит о том, что данный пакет является промежуточным (не последним) фрагментом. Оставшийся бит зарезервирован.

Поле **смещения фрагмента** занимает 13 бит и задает смещение в байтах поля данных этого фрагмента относительно начала поля данных исходного (нефрагментированного) пакета.

Используется при сборке/разборке фрагментов пакетов. Смещение должно быть кратно 8 байт.

Поле **времени жизни** (Time To Live, TTL) занимает один байт и используется для задания предельного срока, в течение которого пакет может перемещаться по сети. Время жизни пакета измеряется в секундах и задается источником. По истечении каждой секунды пребывания на каждом из маршрутизаторов, через которые проходит пакет во время своего «путешествия» по сети, из его текущего времени жизни вычитается единица; единица вычитается и в том случае, если время пребывания было меньше секунды. Поскольку современные маршрутизаторы редко обрабатывают пакет дольше чем за одну секунду, то время жизни можно интерпретировать как максимальное число транзитных узлов, которые разрешено пройти пакету. Если значение поля времени жизни становится нулевым до того, как пакет достигает получателя, пакет уничтожается. Таким образом, время жизни является своего рода часовым механизмом самоуничтожения пакета.

Поле **протокола верхнего уровня** занимает один байт и содержит идентификатор, указывающий, какому протоколу верхнего уровня принадлежит информация, размещенная в поле данных пакета. Значения идентификаторов для разных протоколов приводятся в документе RFC 1700, доступном по адресу http://www.iana.org. Например, 6 означает, что в пакете находится сообщение протокола TCP, 17 — протокола UDP, 1 — протокола ICMP.

Контрольная сумма заголовка занимает 2 байта (16 бит) и рассчитывается только по заголовку. Поскольку некоторые поля заголовка меняют свое значение в процессе передачи пакета по сети (например, поле времени жизни), контрольная сумма проверяется и повторно рассчитывается на каждом маршрутизаторе и конечном узле как дополнение к сумме всех 16-битных слов заголовка. При вычислении контрольной суммы значение самого поля контрольной суммы устанавливается в нуль. Если контрольная сумма неверна, то пакет отбрасывается, как только обнаруживается ошибка.

Поля **IP-адресов источника** и **приемника** имеют одинаковую длину — 32 бита.

Поле **параметров** является не обязательным и используется обычно только при отладке сети. Это поле состоит из нескольких подполей одного из восьми предопределенных типов. В этих подполях можно указывать точный маршрут, по которому маршрутизаторы должны направлять данный пакет (это называется **маршрутизацией от источника**), регистрировать проходимые пакетом маршрутизаторы или помещать данные системы безопасности и временны́е отметки. Так как число подполей в поле параметров может быть произвольным, то в конце заголовка должно быть добавлено несколько нулевых байтов для **выравнивания** заголовка пакета по 32-битной границе.

Далее приведена распечатка значений полей заголовка одного из реальных IP-пакетов, захваченных в сети Ethernet средствами анализатора протоколов сетевого монитора (Network Monitor, NM) компании Microsoft. В данной распечатке NM в скобках дает шестнадцатеричные значения полей, кроме того, программа иногда представляет числовые коды полей в виде, более удобном для чтения. Например, дружественный программный интерфейс NM интерпретирует код 6 в *поле протокола верхнего уровня*, помещая туда название соответствующего протокола — TCP (см. строку, выделенную полужирным шрифтом).

```
IP: Version = 4 (0x4)
IP: Header Length = 20 (0x14)
```

```
IP: Service Type = 0 (0x0)
IP: Precedence = Routine
IP: ...0.... = Normal Delay
IP: ....0... = Normal Throughput
IP: .....0.. = Normal Reliability
IP: Total Length = 54 (0x36)
IP: Identification = 31746 (0x7C02)
IP: Flags Summary = 2 (0x2)
IP: .......0 = Last fragment in datagram
IP: ......1. = Cannot fragment datagram
IP: Fragment Offset = 0 (0x0) bytes
IP: Time to Live = 128 (0x80)
IP: Protocol = TCP - Transmission Control
IP: Checksum = 0xEB86
IP: Source Address = 194.85.135.75
IP: Destination Address = 194.85.135.66
IP: Data: Number of data bytes remaining = 34 (0x0022)
```

Схема IP-маршрутизации

Рассмотрим механизм IP-маршрутизации на примере составной сети, представленной на рис. 15.2. В этой сети 20 маршрутизаторов (изображенных в виде пронумерованных квадрантных блоков) объединяют 18 сетей в общую сеть; N1, N2, ..., N18 — это номера сетей. На каждом маршрутизаторе и конечных узлах *A* и *B* функционируют протоколы IP.

К нескольким интерфейсам (портам) маршрутизаторов присоединяются сети. Каждый интерфейс маршрутизатора можно рассматривать как отдельный узел сети: он имеет сетевой адрес и локальный адрес в той подсети, которая к нему подключена. Например, маршрутизатор под номером 1 имеет три интерфейса, к которым подключены сети N1, N2, N3. На рисунке сетевые адреса этих портов обозначены IP_{11}, IP_{12} и IP_{13}. Интерфейс IP_{11} является узлом сети N1, и следовательно, в поле номера сети порта IP_{11} содержится номер N1. Аналогично интерфейс IP_{12} — это узел в сети N2, а порт IP_{13} — узел в сети N3. Таким образом, маршрутизатор можно рассматривать как совокупность нескольких узлов, каждый из которых входит в свою сеть. Как единое устройство маршрутизатор не имеет выделенного адреса, ни сетевого, ни локального.

ПРИМЕЧАНИЕ

При наличии у маршрутизатора блока управления (например, по протоколу SNMP) этот блок имеет собственные локальный и сетевой адреса, по которым к нему обращается центральная станция управления. Эти адреса выбираются из того же пула, что и адреса физических интерфейсов маршрутизатора. В технической документации такого рода адреса называются адресами обратной петли (looopback address), или адресами виртуальных интерфейсов (virtual interface address). В отличие от адресов 127.x.x.x, зарезервированных для передачи данных между программными компонентами, находящимися в пределах одного компьютера, адреса виртуальных интерфейсов предполагают обращение к ним извне.

В сложных составных сетях почти всегда существуют несколько альтернативных маршрутов для передачи пакетов между двумя конечными узлами. Так, пакет, отправленный из узла *A* в узел *B*, может пройти через маршрутизаторы 17, 12, 5, 4 и 1 или маршрутизаторы 17, 13, 7, 6 и 3. Нетрудно найти еще несколько маршрутов между узлами *A* и *B*.

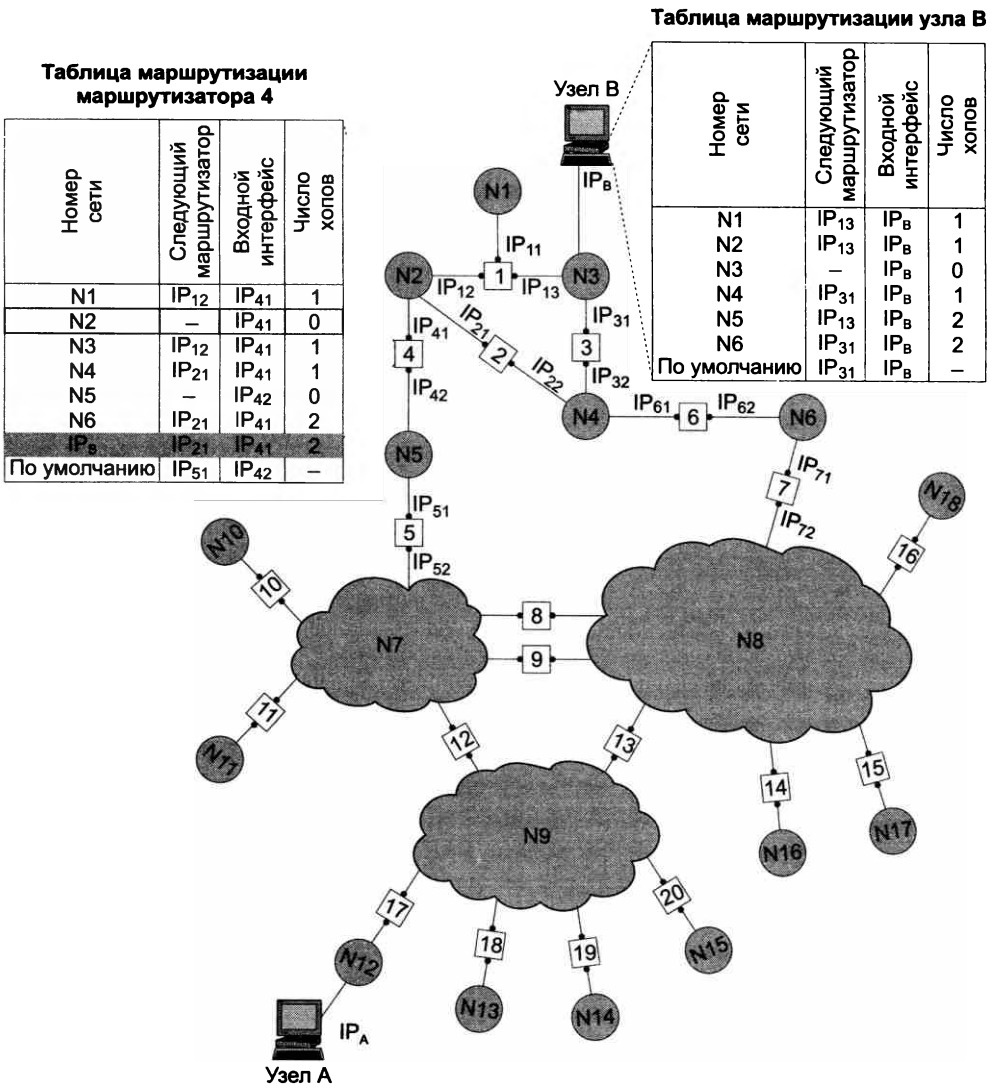


Рис. 15.2. Принципы маршрутизации в составной сети

Задачу выбора маршрута из нескольких возможных решают маршрутизаторы, а также конечные узлы. Маршрут выбирается на основании имеющейся у этих устройств информации о текущей конфигурации сети, а также на основании критерия выбора маршрута. В качестве критерия часто выступает задержка прохождения маршрута отдельным пакетом, средняя пропускная способность маршрута для последовательности пакетов или наиболее простой критерий, учитывающий только количество пройденных на маршруте промежуточных маршрутизаторов (*ретрансляционных участков*, или *хопов*). Полученная в результате анализа информация о маршрутах дальнейшего следования пакетов помещается в **таблицу маршрутизации**.

Упрощенная таблица маршрутизации

Используя условные обозначения для сетевых адресов маршрутизаторов и номеров сетей, показанные на рис. 15.2, посмотрим, как могла бы выглядеть таблица маршрутизации, например, в маршрутизаторе 4 (табл. 15.1).

Таблица 15.1. Таблица маршрутизации маршрутизатора 4

Адрес назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{12} (R1)	IP41	1
N2	–	IP41	0 (подсоединена)
N3	IP_{12} (R1)	IP41	1
N4	IP_{21} (R2)	IP41	1
N5	–	IP42	0 (подсоединена)
N6	IP_{21} (R2)	IP21	2
IP_B	IP_{21} (R2)	IP41	2
Маршрут по умолчанию	IP_{51} (R5)	IP42	–

Первый столбец таблицы содержит **адреса назначения пакетов**.

В каждой строке таблицы следом за адресом назначения указывается **сетевой адрес следующего маршрутизатора** (точнее, сетевой адрес интерфейса следующего маршрутизатора), на который надо направить пакет, чтобы тот передвигался по направлению к заданному адресу по рациональному маршруту.

Перед тем как передать пакет следующему маршрутизатору, текущий маршрутизатор должен определить, на какой из нескольких собственных портов (IP_{41} или IP_{42}) он должен поместить данный пакет. Для этого служит третий столбец таблицы маршрутизации, содержащий **сетевые адреса выходных интерфейсов**.

Некоторые реализации сетевых протоколов допускают наличие в таблице маршрутизации сразу *нескольких строк*, соответствующих одному и тому же адресу назначения. В этом случае при выборе маршрута принимается во внимание столбец, представляющий расстояние до сети назначения. При этом расстояние измеряется в любой метрике, используемой в соответствии с заданным в сетевом пакете критерием. Расстояние может измеряться временем прохождения пакета по линиям связи, различными характеристиками надежности линий связи на данном маршруте, пропускной способностью или другой величиной, отражающей качество данного маршрута по отношению к заданному критерию. В табл. 15.1 расстояние между сетями измеряется хопами. Расстояние для сетей, непосредственно подключенных к портам маршрутизатора, здесь принимается равным 0, однако в некоторых реализациях отсчет расстояний начинается с 1.

Когда пакет поступает на маршрутизатор, модуль IP извлекает из его заголовка номер сети назначения и последовательно сравнивает его с номерами сетей из каждой строки таблицы. Строка с совпавшим номером сети показывает ближайший маршрутизатор, на который следует направить пакет. Например, если на какой-либо порт маршрутизатора 4 поступает пакет, адресованный в сеть N6, то из таблицы маршрутизации следует, что адрес

следующего маршрутизатора — IP_{21}, то есть очередным этапом движения данного пакета будет движение к порту 1 маршрутизатора 2.

Чаще всего в качестве адреса назначения в таблице указывается не весь IP-адрес, а только номер сети назначения. Таким образом, для всех пакетов, направляемых в одну и ту же сеть, протокол IP будет предлагать один и тот же маршрут (мы пока не принимаем во внимание возможные изменения состояния сети, такие как отказы маршрутизаторов или обрывы кабелей). Однако в некоторых случаях возникает необходимость для одного из узлов сети определить **специфический маршрут**, отличающийся от маршрута, заданного для всех остальных узлов сети. Для этого в таблицу маршрутизации помещают для данного узла отдельную строку, содержащую его полный IP-адрес и соответствующую маршрутную информацию. Такого рода запись имеется в табл. 15.1 для узла *B*. Пусть, например, администратор маршрутизатора 4, руководствуясь соображениями безопасности, решил, что пакеты, следующие в узел *B* (полный адрес IP_B), должны идти через маршрутизатор 2 (интерфейс IP_{21}), а не маршрутизатор 1 (интерфейс IP_{12}), через который передаются пакеты всем остальным узлам сети N3. Если в таблице имеются записи о маршрутах как к сети в целом, так и к ее отдельному узлу, то при поступлении пакета, адресованного данному узлу, маршрутизатор отдаст предпочтение специфическому маршруту.

Поскольку пакет может быть адресован *в любую сеть* составной сети, может показаться, что каждая таблица маршрутизации должна иметь записи обо *всех* сетях, входящих в составную сеть. Однако при таком подходе в случае крупной сети объем таблиц маршрутизации может оказаться очень большим, что повлияет на время ее просмотра, потребует много места для хранения и т. п. Поэтому на практике широко известен прием уменьшения количества записей в таблице маршрутизации, основанный на введении **маршрута по умолчанию** (default route), учитывающего особенности топологии сети. Рассмотрим, например, маршрутизаторы, находящиеся на периферии составной сети. В их таблицах достаточно записать номера только тех сетей, которые непосредственно подсоединены к данному маршрутизатору или расположены поблизости на тупиковых маршрутах. Обо всех же остальных сетях можно сделать в таблице единственную запись, указывающую на маршрутизатор, через который пролегает путь ко всем этим сетям. Такой маршрутизатор называется **маршрутизатором по умолчанию** (default router). В нашем примере на маршрутизаторе 4 имеются специфические маршруты только для пакетов, следующих в сети N1–N6. Для всех остальных пакетов, адресованных в сети N7–N18, маршрутизатор предлагает продолжить путь через один и тот же порт IP_{51} маршрутизатора 5, который в данном случае и является маршрутизатором по умолчанию.

Таблицы маршрутизации конечных узлов

Задачу маршрутизации решают не только промежуточные узлы (маршрутизаторы), но и конечные узлы — компьютеры. Решение этой задачи начинается с того, что средствами протокола IP на конечном узле определяется, направлен ли пакет в другую сеть или адресован какому-нибудь узлу данной сети. Если номер сети назначения совпадает с номером данной сети, это означает, что пакет маршрутизировать не требуется. В противном случае маршрутизация нужна.

Структуры таблиц маршрутизации конечных узлов и транзитных маршрутизаторов аналогичны. Обратимся снова к сети, изображенной на рис. 15.2. Таблица маршрутизации конечного узла *B*, принадлежащего сети N3, могла бы выглядеть так, как табл. 15.2. Здесь

IP_B — сетевой адрес интерфейса компьютера *B*. На основании этой таблицы конечный узел *B* выбирает, на какой из двух имеющихся в локальной сети N3 маршрутизаторов (R1 или R3) следует посылать тот или иной пакет.

Таблица 15.2. Таблица маршрутизации конечного узла B

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{13} (R1)	IP_B	1
N2	IP_{13} (R1)	IP_B	1
N3	—	IP_B	0
N4	IP_{31} (R3)	IP_B	1
N5	IP_{13} (R1)	IP_B	2
N6	IP_{31} (R3)	IP_B	2
Маршрут по умолчанию	IP_{31} (R3)	IP_B	—

Конечные узлы в еще большей степени, чем маршрутизаторы, пользуются приемом маршрутизации по умолчанию. Хотя они также в общем случае имеют в своем распоряжении таблицу маршрутизации, ее объем обычно незначителен, что объясняется периферийным расположением всех конечных узлов. Конечный узел часто вообще работает без таблицы маршрутизации, имея только сведения об адресе маршрутизатора по умолчанию. При наличии одного маршрутизатора в локальной сети этот вариант — единственно возможный для всех конечных узлов. Но даже при наличии нескольких маршрутизаторов в локальной сети, когда перед конечным узлом стоит проблема их выбора, часто в компьютерах для повышения производительности прибегают к заданию маршрута по умолчанию.

Рассмотрим таблицу маршрутизации другого конечного узла составной сети — узла *A* (табл. 15.3). Компактный вид таблицы маршрутизации узла *A* отражает тот факт, что все пакеты, направляемые из узла *A*, либо не выходят за пределы сети N12, либо непременно проходят через порт 1 маршрутизатора 17. Этот маршрутизатор и определен в таблице маршрутизации в качестве маршрутизатора по умолчанию.

Таблица 15.3. Таблица маршрутизации конечного узла A

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N12	—	IPA	0
Маршрут по умолчанию	$IP_{17,1}$ (R17)	IPA	—

Еще одним отличием работы маршрутизатора и конечного узла является способ построения таблицы маршрутизации. Если маршрутизаторы, как правило, автоматически создают таблицы маршрутизации, обмениваясь служебной информацией, то для конечных узлов таблицы маршрутизации часто создаются вручную администраторами и хранятся в виде постоянных файлов на дисках.

Просмотр таблиц маршрутизации без масок

Рассмотрим алгоритм просмотра таблицы маршрутизации, реализуемый на маршрутизаторе протоколом IP. При его описании мы будем использовать табл. 15.1 и рис. 15.2.

1. Пусть на один из интерфейсов маршрутизатора поступает пакет. Протокол IP извлекает из пакета IP-адрес назначения (предположим, адрес назначения IP_B).
2. Выполняется *первая фаза* просмотра таблицы — *поиск конкретного маршрута к узлу*. IP-адрес (целиком) последовательно строка за строкой сравнивается с содержимым поля адреса назначения таблицы маршрутизации. Если произошло совпадение (как в табл. 15.1), то из соответствующей строки извлекаются адрес следующего маршрутизатора (IP_{21}) и идентификатор выходного интерфейса (IP_{41}). На этом просмотр таблицы заканчивается.
3. Предположим теперь, что в таблице нет строки с адресом назначения IP_B, а значит, совпадения не произошло. В этом случае протокол IP переходит ко *второй фазе* просмотра — *поиску маршрута к сети назначения*. Из IP-адреса выделяется номер сети (в нашем примере из адреса IP_B выделяется номер сети N3), и таблица снова просматривается на предмет совпадения номера сети в какой-либо строке с номером сети из пакета. При совпадении (а в нашем примере оно произошло) из соответствующей строки таблицы извлекаются адрес следующего маршрутизатора (IP_{12}) и идентификатор выходного интерфейса (IP_{41}). Просмотр таблицы на этом завершается.
4. Наконец, предположим, что адрес назначения в пакете был таков, что совпадения не произошло ни в первой, ни во второй фазе просмотра. В таком случае средствами протокола IP либо выбирается маршрут по умолчанию (и пакет направляется по адресу IP_{51}), либо, если маршрут по умолчанию отсутствует, пакет отбрасывается[1]. Просмотр таблицы на этом заканчивается.

ВНИМАНИЕ

Последовательность фаз в данном алгоритме строго определена, в то время как последовательность просмотра, или, что одно и одно же, порядок расположения строк в таблице, включая запись о маршруте по умолчанию, никак не сказывается на результате.

Примеры таблиц маршрутизации разных форматов

Структура реальных таблиц маршрутизации стека TCP/IP в целом соответствует упрощенной структуре рассмотренных ранее таблиц. Отметим, однако, что вид таблицы IP-маршрутизации зависит от конкретной реализации стека TCP/IP. Приведем пример нескольких вариантов таблицы маршрутизации, с которыми мог бы работать маршрутизатор R1 в сети, представленной на рис. 15.3.

---

[1] Стандарты технологии TCP/IP не требуют, чтобы в таблице маршрутизации непременно содержались маршруты для всех пакетов, которые могут прийти на его интерфейсы, более того, в таблице может отсутствовать маршрут по умолчанию.

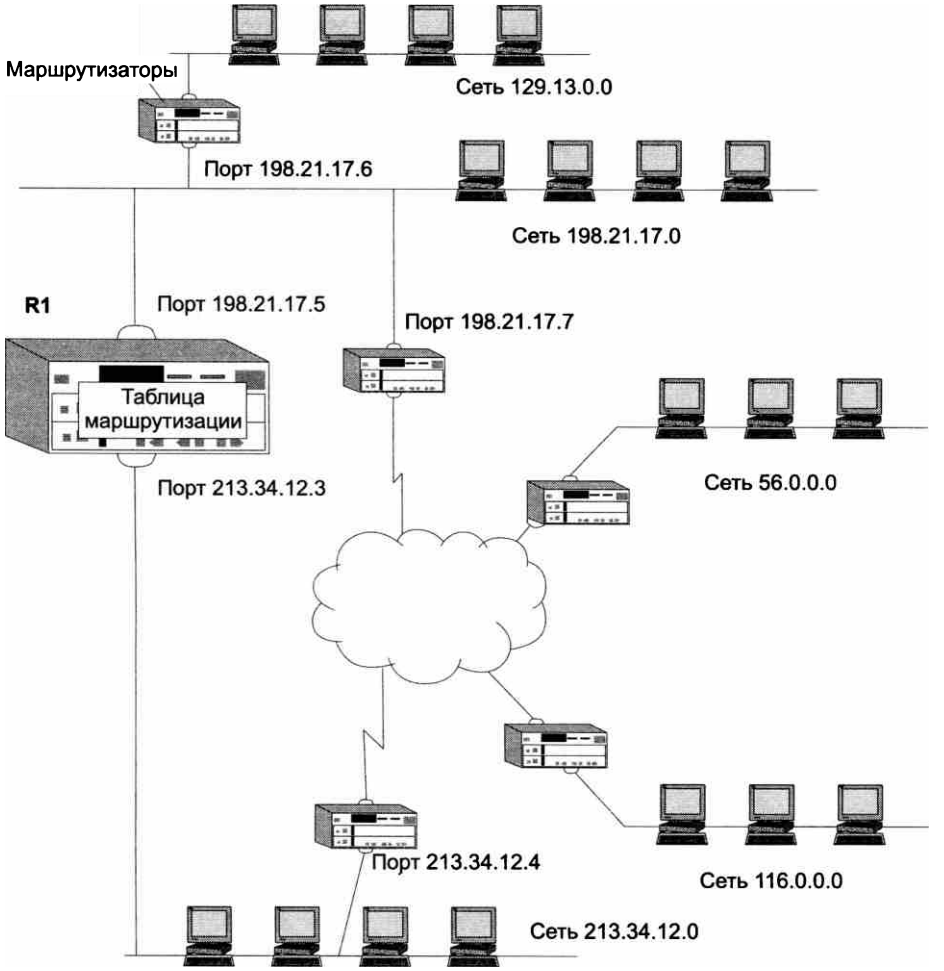


Рис. 15.3. Пример маршрутизируемой сети

Начнем с «придуманного» предельно упрощенного варианта таблицы маршрутизации (табл. 15.4). Здесь имеются три маршрута к сетям (записи 56.0.0.0, 116.0.0.0 и 129.13.0.0), две записи о непосредственно подсоединенных сетях (198.21.17.0 и 213.34.12.0), а также запись о маршруте по умолчанию.

Таблица 15.4. Упрощенная таблица маршрутизации маршрутизатора R1

Адрес сети назначения	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние до сети назначения
56.0.0.0	213.34.12.4	213.34.12.3	15
116.0.0.0	213.34.12.4	213.34.12.3	13
129.13.0.0	198.21.17.6	198.21.17.5	2
198.21.17.0	198.21.17.5	198.21.17.5	1 (подсоединена)

Адрес сети назначения	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние до сети назначения
213.34.12.0	213.34.12.3	213.34.12.3	1 (подсоединена)
Маршрут по умолчанию	198.21.17.7	198.21.17.5	–

Более сложный вид имеют таблицы, которые генерируются в промышленно выпускаемом сетевом оборудовании.

Если представить, что в качестве маршрутизатора R1 в данной сети работает штатный *программный* маршрутизатор операционной системы Microsoft Windows, то его таблица маршрутизации могла бы выглядеть так, как табл. 15.5.

Таблица 15.5. Таблица программного маршрутизатора ОС Windows

Сетевой адрес	Маска	Адрес шлюза	Интерфейс	Метрика
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
0.0.0.0	0.0.0.0	198.21.17.7	198.21.17.5	1
56.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	15
116.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	13
129.13.0.0	255.255.0.0	198.21.17.6	198.21.17.5	2
198.21.17.0	255.255.255.0	198.21.17.5	198.21.17.5	1
198.21.17.5	255.255.255.255	127.0.0.1	127.0.0.1	1
198.21.17.255	255.255.255.255	198.21.17.5	198.21.17.5	1
213.34.12.0	255.255.255.0	213.34.12.3	213.34.12.3	1
213.34.12.3	255.255.255.255	127.0.0.1	127.0.0.1	1
213.34.12.255	255.255.255.255	213.34.12.3	213.34.12.3	1
224.0.0.0	224.0.0.0	198.21.17.6	198.21.17.6	1
224.0.0.0	224.0.0.0	213.34.12.3	213.34.12.3	1
255.255.255.255	255.255.255.255	198.21.17.6	198.21.17.6	1

Если на месте маршрутизатора R1 установить один из популярных *аппаратных* маршрутизаторов, то его таблица маршрутизации для этой же сети может выглядеть совсем иначе (табл. 15.6).

Таблица 15.6. Таблица маршрутизации аппаратного маршрутизатора

Адрес назначения	Маска	Шлюз	Метрика	Статус	TTL	Источник
198.21.17.0	255.255.255.0	198.21.17.5	0	Up	–	Подключена
213.34.12.0	255.255.255.0	213.34.12.3	0	Up	–	Подключена
56.0.0.0	255.0.0.0	213.34.12.4	14	Up	–	Статичсская
116.0.0.0	255.0.0.0	213.34.12.4	12	Up	–	Статическая
129.13.0.0	255.255.0.0	198.21.17.6	1	Up	160	RIP

И наконец, табл. 15.7 представляет собой таблицу маршрутизации для того же маршрутизатора R1, реализованного в виде программного маршрутизатора одной из версий операционной системы Unix.

Таблица 15.7. Таблица маршрутизации маршрутизатора Unix

Адрес назначения	Шлюз	Флаги	Число ссылок	Загрузка	Интерфейс
127.0.0.0	127.0.0.1	UH	1	154	lo0
Маршрут по умолчанию	198.21.17.7	UG	5	43270	le0
198.21.17.0	198.21.17.5	U	35	246876	le0
213.34.12.0	213.34.12.3	U	44	132435	le1
129.13.0.0	198.21.1.7.6	UG	6	16450	le0
56.0.0.0	213.34.12.4	UG	12	5764	le1
116.0.0.0	213.34.12.4	UG	21	23544	le1

ПРИМЕЧАНИЕ

Заметим, что поскольку между структурой сети и таблицей маршрутизации нет однозначного соответствия, для каждого из приведенных вариантов таблицы можно предложить свои «подварианты», отличающиеся выбранным маршрутом к той или иной сети. В данном случае внимание концентрируется на существенных различиях в форме представления маршрутной информации разными реализациями маршрутизаторов.

Несмотря на достаточно заметные внешние различия, во всех трех «реальных» таблицах присутствуют все ключевые данные из рассмотренной упрощенной таблицы, без которых невозможна маршрутизация пакетов.

К таким данным, во-первых, относятся *адреса сети назначения* (столбцы «Адрес назначения» в аппаратном маршрутизаторе и маршрутизаторе Unix или столбец «Сетевой адрес» в маршрутизаторе ОС Windows).

Вторым обязательным полем таблицы маршрутизации является *адрес следующего маршрутизатора* (столбцы «Шлюз» в аппаратном маршрутизаторе и маршрутизаторе Unix или столбец «Адрес шлюза» в маршрутизаторе ОС Windows).

Третий ключевой параметр — *адрес порта*, на который нужно направить пакет, в некоторых таблицах указывается прямо (столбец «Интерфейс» в таблице маршрутизатора ОС Windows), а в некоторых — косвенно. Так, в таблице маршрутизатора Unix вместо адреса порта задается его условное наименование — le0 для порта с адресом 198.21.17.5, le1 для порта с адресом 213.34.12.3 и lo0 для внутреннего порта с адресом 127.0.0.1. В аппаратном маршрутизаторе поле, обозначающее выходной порт в какой-либо форме, вообще отсутствует. Это объясняется тем, что адрес выходного порта всегда можно косвенно определить по адресу следующего маршрутизатора. Например, определим по табл. 15.6 адрес выходного порта для сети 56.0.0.0. Из таблицы следует, что следующим маршрутизатором для этой сети будет маршрутизатор с адресом 213.34.12.4. Адрес следующего маршрутизатора должен принадлежать одной из непосредственно присоединенных к маршрутизатору сетей,

и в данном случае это сеть 213.34.12.0. Маршрутизатор имеет порт, присоединенный к этой сети, и адрес этого порта 213.34.12.3 мы находим в столбце «Шлюз» второй строки таблицы маршрутизации, которая описывает непосредственно присоединенную сеть 213.34.12.0. Для непосредственно присоединенных сетей адресом следующего маршрутизатора всегда является адрес собственного порта маршрутизатора. Таким образом, для сети 56.0.0 адресом выходного порта является 213.34.12.3.

Стандартным решением сегодня является использование поля маски в каждой записи таблицы, как это сделано в таблицах маршрутизатора ОС Windows и аппаратного маршрутизатора (столбцы «Маска»). Механизм обработки масок при принятии решения маршрутизаторами рассматривается далее. Отсутствие поля маски говорит о том, что либо маршрутизатор рассчитан на работу только с тремя стандартными классами адресов, либо для всех записей используется одна и та же маска, что снижает гибкость маршрутизации.

Поскольку в таблице маршрутизации маршрутизатора Unix каждая сеть назначения упомянута только один раз, а значит, возможность выбора маршрута отсутствует, то поле метрики является не обязательным параметром. В остальных двух таблицах поле метрики используется только для указания на то, что сеть подключена непосредственно. Метрика 0 для аппаратного маршрутизатора или 1 для маршрутизатора ОС Windows говорит маршрутизатору, что эта сеть непосредственно подключена к его порту, а другое значение метрики соответствует удаленной сети. Выбор метрики для непосредственно подключенной сети (1 или 0) является произвольным, главное, чтобы метрика удаленной сети отсчитывалась с учетом этого выбранного начального значения. В маршрутизаторе Unix используется поле признаков, где флаг G (Gateway — шлюз) отмечает удаленную сеть, а его отсутствие — непосредственно подключенную.

Признак непосредственно подключенной сети говорит маршрутизатору, что пакет уже достиг своей сети, поэтому протокол IP активизирует ARP-запрос относительно IP-адреса *узла назначения*, а не следующего маршрутизатора.

Однако существуют ситуации, когда маршрутизатор должен обязательно хранить значение метрики для записи о каждой удаленной сети. Эти ситуации возникают, когда записи в таблице маршрутизации являются результатом работы некоторых протоколов маршрутизации, например протокола RIP. В таких протоколах новая информация о какой-либо удаленной сети сравнивается с информацией, содержащейся в таблице в данный момент, и если значение новой метрики лучше текущей, то новая запись вытесняет имеющуюся. В таблице маршрутизатора Unix поле метрики отсутствует, и это значит, что он не использует протокол RIP.

Флаги записей присутствуют только в таблице маршрутизатора Unix.

- U — маршрут активен и работоспособен. Аналогичный смысл имеет поле статуса в аппаратном маршрутизаторе.
- H — признак специфического маршрута к определенному хосту.
- G — маршрут пакета проходит через промежуточный маршрутизатор (шлюз). Отсутствие этого флага отмечает непосредственно подключенную сеть.
- D — маршрут получен из перенаправленного сообщения протокола ICMP. Этот признак может присутствовать только в таблице маршрутизации *конечного узла*. Признак означает, что конечный узел при какой-то предыдущей передаче пакета выбрал не самый

рациональный следующий маршрутизатор на пути к данной сети и этот маршрутизатор с помощью протокола ICMP сообщил конечному узлу, что все последующие пакеты к данной сети нужно отправлять через другой маршрутизатор.

В таблице маршрутизатора Unix используются еще два поля, имеющих справочное значение. Поле числа ссылок показывает, сколько раз на данный маршрут ссылались при продвижении пакетов. Поле загрузки отражает количество байтов, переданных по данному маршруту.

В записях таблиц аппаратного маршрутизатора также имеются два справочных поля. Поле **времени жизни записи** (TTL) в данном случае никак не связано со временем жизни пакета. Здесь оно показывает время, в течение которого значение данной записи еще действительно. **Поле источника** говорит об источнике появления записи в таблице маршрутизации.

Источники и типы записей в таблице маршрутизации

Практически для всех маршрутизаторов существуют *три* основных источника записей в таблице.

- Одним из источников записей в таблице маршрутизации является **программное обеспечение стека TCP/IP**, которое при инициализации маршрутизатора автоматически заносит в таблицу несколько записей, в результате чего создается так называемая минимальная таблица маршрутизации. Программное обеспечение формирует записи о *непосредственно подключенных сетях* и маршрутах по умолчанию, информация о которых появляется в стеке при ручном конфигурировании интерфейсов компьютера или маршрутизатора. К таким записям в приведенных примерах относятся записи о сетях 213.34.12.0 и 198.21.17.0, а также запись о маршруте по умолчанию в маршрутизаторе Unix и запись 0.0.0.0 в маршрутизаторе ОС Windows. Кроме того, программное обеспечение автоматически заносит в таблицу маршрутизации записи *об адресах особого назначения*. В приведенных примерах таблица маршрутизатора ОС Windows содержит наиболее полный набор записей такого рода. Несколько записей в этой таблице связано с особым адресом 127.0.0.0. Записи с адресом 224.0.0.0 требуются для обработки групповых адресов. Кроме того, в таблицу могут быть занесены адреса, предназначенные для обработки широковещательных рассылок (например, записи 8 и 11 содержат адрес отправки широковещательного сообщения в соответствующих подсетях, а последняя запись в таблице — адрес ограниченной широковещательной рассылки). Заметим, что в некоторых таблицах записи об особых адресах вообще отсутствуют.
- Еще одним источником записей в таблице является **администратор**, непосредственно формирующий записи с помощью некоторой системной утилиты, например программы route, доступной в операционных системах Unix и Windows. В аппаратных маршрутизаторах также всегда имеется команда для ручного задания записей таблицы маршрутизации. Заданные вручную записи всегда являются *статическими*, то есть они не имеют срока жизни. Эти записи могут быть как постоянными, то есть сохраняющимися при перезагрузке маршрутизатора, так и временными, хранящимися в таблице только до выключения устройства. Часто администратор вручную заносит запись о маршруте по умолчанию. Таким же образом в таблицу маршрутизации может быть внесена запись о специфическом для узла маршруте.

- И наконец, третьим источником записей могут быть **протоколы маршрутизации**, такие как RIP или OSPF. Эти записи всегда являются *динамическими*, то есть имеют ограниченный срок жизни.

Программные маршрутизаторы Windows и Unix не показывают источник появления той или иной записи в таблице, а аппаратный маршрутизатор использует для этой цели поле источника. В приведенном в табл. 15.6 примере первые две записи созданы программным обеспечением стека на основании данных о конфигурации портов маршрутизатора — это показывает признак «Подключена». Следующие две записи обозначены как статические — это означает, что их ввел вручную администратор. Последняя запись является следствием работы протокола RIP, поэтому в ее поле «TTL» имеется значение 160.

Пример IP-маршрутизации без масок

Рассмотрим процесс продвижения пакета в составной сети на примере IP-сети, показанной на рис. 15.4. При этом будем считать, что все узлы сети, рассматриваемой в примере, имеют *адреса, основанные на классах*. Особое внимание будет уделено взаимодействию протокола IP с протоколами разрешения адресов ARP и DNS.

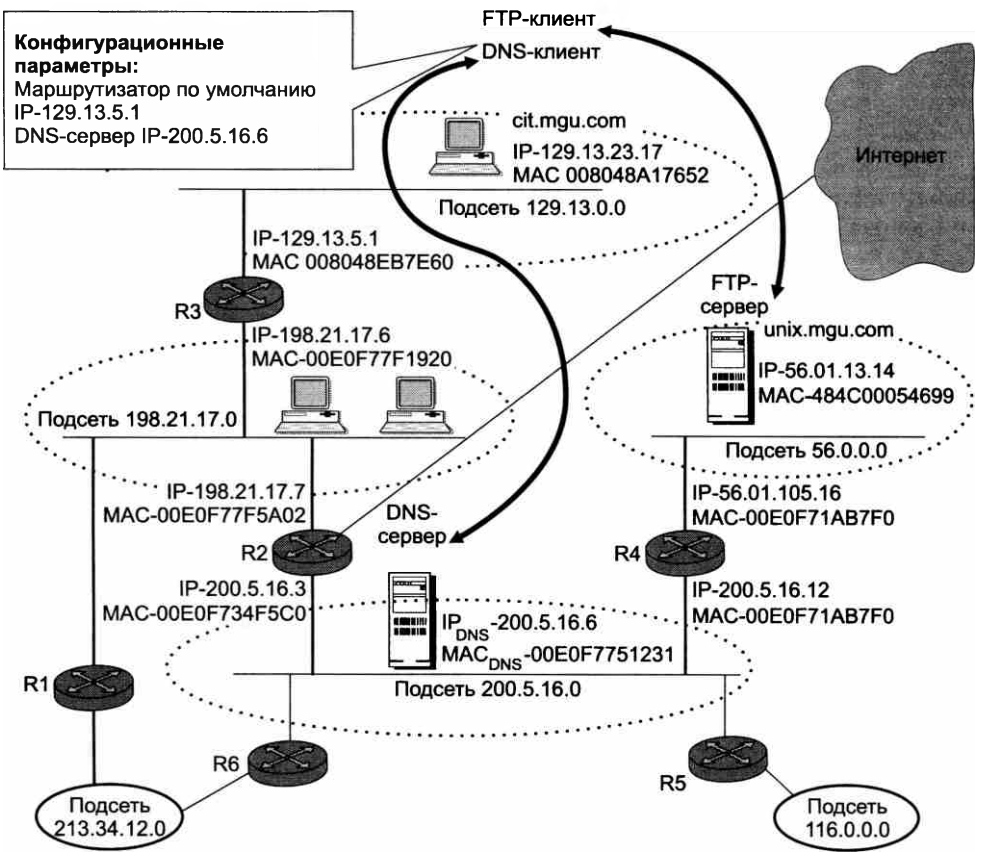


Рис. 15.4. Пример IP-маршрутизации

Итак, пусть пользователю компьютера cit.mgu.com, находящегося в сети 129.13.0.0, необходимо установить связь с FTP-сервером. Пользователю известно символьное имя сервера unix.mgu.com, поэтому он набирает на клавиатуре команду обращения к FTP-серверу по имени:

```
> ftp unix.mgu.com
```

Выполнение этой команды инициирует три последовательные операции:

1. DNS-клиент (работающий на компьютере cit.mgu.com) передает DNS-серверу сообщение, в котором содержится запрос об IP-адресе сервера unix.mgu.com, с которым он хочет связаться по протоколу FTP.
2. DNS-сервер, выполнив поиск, передает ответ DNS-клиенту о найденном IP-адресе сервера unix.mgu.com.
3. FTP-клиент (работающий на том же компьютере cit.mgu.com), используя найденный IP-адрес сервера unix.mgu.com, передает сообщение работающему на нем FTP-серверу.

Давайте последовательно, по шагам, рассмотрим, как при решении этих задач взаимодействуют между собой протоколы DNS, IP, ARP и Ethernet и что происходит при этом с кадрами и пакетами.

1. *Формирование IP-пакета с инкапсулированным в него DNS-запросом.* Программный модуль FTP-клиента, получив команду `> ftp unix.mgu.com`, передает запрос к работающей на этом же компьютере клиентской части протокола DNS, которая, в свою очередь, формирует к DNS-серверу запрос, интерпретируемый примерно так: «Какой IP-адрес соответствует символьному имени unix.mgu.com?» Запрос упаковывается в UDP-дейтаграмму, затем в IP-пакет. В заголовке пакета в качестве адреса назначения указывается IP-адрес 200.5.16.6 DNS-сервера. Этот адрес известен программному обеспечению клиентского компьютера, так как он входит в число его конфигурационных параметров. Сформированный IP-пакет будет перемещаться по сети в неизменном виде (как показано на рис. 15.5), пока не дойдет до адресата — DNS-сервера.

Рис. 15.5. IP-пакет с инкапсулированным в него DNS-запросом

2. *Передача кадра Ethernet с IP-пакетом маршрутизатору R3.* Для передачи этого IP-пакета необходимо его упаковать в кадр Ethernet, указав в заголовке MAC-адрес получателя. Технология Ethernet способна доставлять кадры только тем адресатам, которые находятся в пределах одной подсети с отправителем. Если же адресат расположен вне этой подсети, то кадр надо передать ближайшему маршрутизатору, чтобы тот взял на себя заботу о дальнейшем перемещении пакета. Для этого модуль IP, сравнив номера сетей в адресах отправителя и получателя, то есть 129.13.23.17 и 200.5.16.6, выясняет, что пакет направляется в другую сеть, следовательно, его необходимо передать маршрутизатору, в данном случае маршрутизатору по умолчанию. IP-адрес маршрутиза-

тора по умолчанию также известен клиентскому узлу, поскольку он входит в число конфигурационных параметров. Однако для кадра Ethernet необходимо указать не IP-адрес, а MAC-адрес получателя. Эта проблема решается с помощью протокола ARP, который для ответа на вопрос: «Какой MAC-адрес соответствует IP-адресу 129.13.5.1?» — делает поиск в своей ARP-таблице. Поскольку обращения к маршрутизатору происходят часто, будем считать, что нужный MAC-адрес обнаруживается в таблице и имеет значение 008048EB7E60. После получения этой информации клиентский компьютер cit.mgu.com отправляет маршрутизатору R3 пакет, упакованный в кадр Ethernet (рис. 15.6).

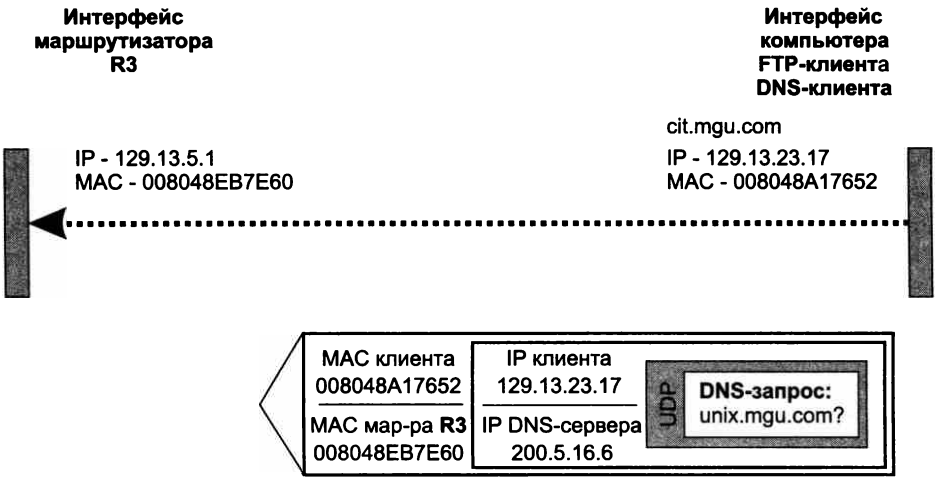


Рис. 15.6. Кадр Ethernet с инкапсулированным IP-пакетом, отправленный с клиентского компьютера

3. *Определение IP-адреса и MAC-адреса следующего маршрутизатора R2.* Кадр принимается интерфейсом 129.13.5.1 маршрутизатора R3. Протокол Ethernet, работающий на этом интерфейсе, извлекает из этого кадра IP-пакет и передает его протоколу IP. Протокол IP находит в заголовке пакета адрес назначения 200.5.16.6 и просматривает записи своей таблицы маршрутизации. Пусть маршрутизатор R3 не обнаруживает специфического маршрута для адреса назначения 200.5.16.6, но находит в своей таблице следующую запись:

   200.5.16.0 198.21.17.7 198.21.17.6

   Эта запись говорит о том, что пакеты для сети 200.5.16.0 маршрутизатор R3 должен передавать на свой выходной интерфейс 198.21.17.6, с которого они поступят на интерфейс следующего маршрутизатора R2, имеющего IP-адрес 198.21.17.7. Однако знания IP-адреса недостаточно, чтобы передать пакет по сети Ethernet. Необходимо определить MAC-адрес маршрутизатора R2. Как известно, такой работой занимается протокол ARP. Пусть на этот раз в ARP-таблице нет записи об адресе маршрутизатора R2. Тогда в сеть отправляется широковещательный ARP-запрос, который поступает на все интерфейсы сети 198.21.17.0. Ответ приходит только от интерфейса маршрутизатора R2: «Я имею IP-адрес 198.21.17.7 и мой MAC-адрес 00E0F77F5A02» (рис. 15.7).

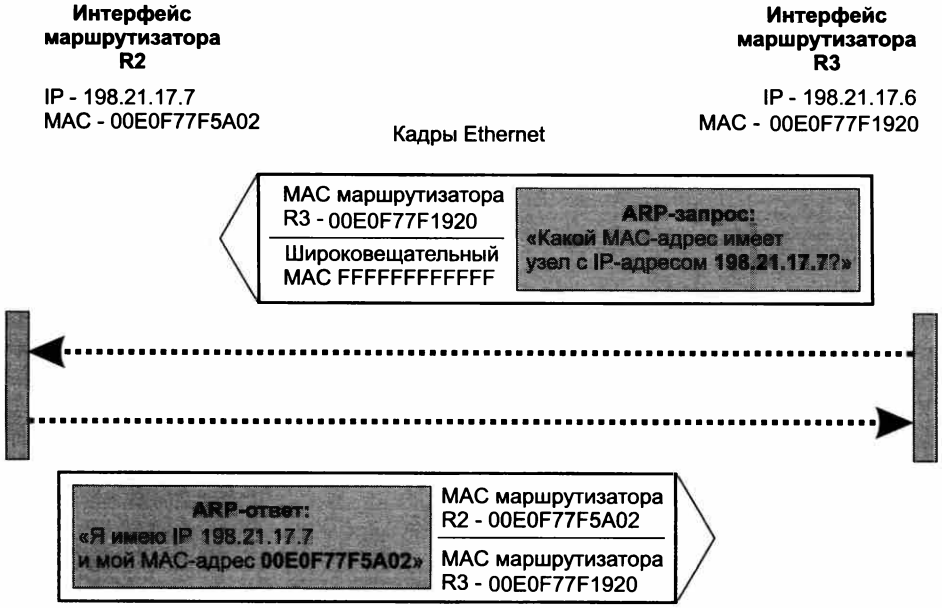


Рис. 15.7. Кадры Ethernet с инкапсулированными ARP-запросом и ARP-ответом

Теперь, зная MAC-адрес маршрутизатора R2 (00E0F77F5A02), маршрутизатор R3 отсылает ему IP-пакет с DNS-запросом (рис. 15.8).

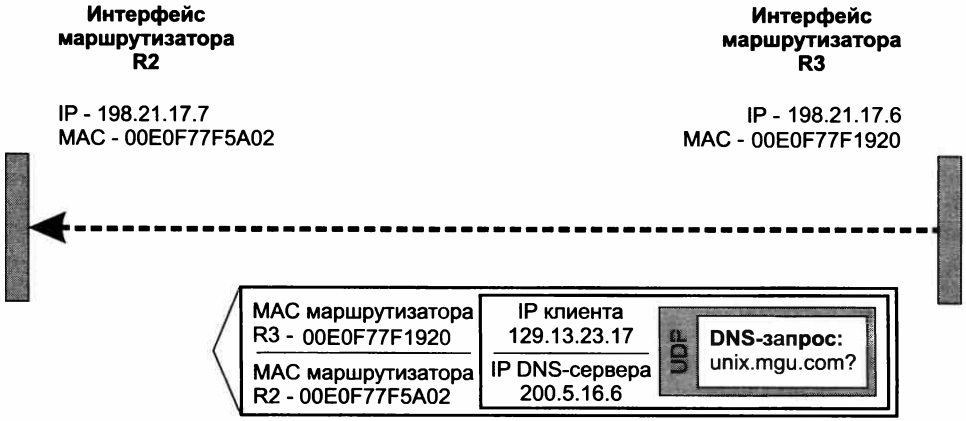


Рис. 15.8. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R3 маршрутизатору R2

4. *Маршрутизатор R2 доставляет пакет DNS-серверу.* Модуль IP на маршрутизаторе R2 действует в соответствии с уже не раз описанной нами процедурой: отбросив заголовок кадра Ethernet, он извлекает из пакета IP-адрес назначения и просматривает свою таблицу маршрутизации. Там он обнаруживает, что сеть назначения 200.5.16.0

является непосредственно присоединенной к его второму интерфейсу. Следовательно, пакет не нужно маршрутизировать, однако требуется определить MAC-адрес узла назначения. Протокол ARP «по просьбе» протокола IP находит (либо в ARP-таблице, либо по запросу) требуемый MAC-адрес 00E0F7751231 DNS-сервера. Получив ответ о MAC-адресе, маршрутизатор R2 отправляет в сеть назначения кадр Ethernet с DNS-запросом (рис. 15.9).

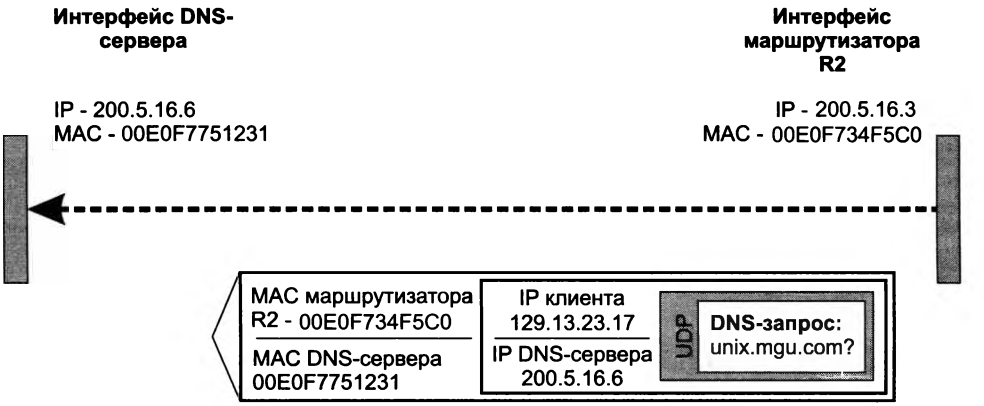


Рис. 15.9. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R2

5. *Сетевой адаптер DNS-сервера захватывает кадр Ethernet*, обнаруживает совпадение MAC-адреса назначения, содержащегося в заголовке, со своим собственным адресом и направляет его модулю IP. После анализа полей заголовка IP из пакета извлекаются данные вышележащих протоколов. DNS-запрос передается программному модулю DNS-сервера. DNS-сервер просматривает свои таблицы, возможно, обращается к другим DNS-серверам и в результате формирует ответ, смысл которого состоит в следующем: «Символьному имени unix.mgu.com соответствует IP-адрес 56.01.13.14».

ПРИМЕЧАНИЕ

Заметим, что во время всего путешествия пакета по составной сети от клиентского компьютера до DNS-сервера IP-адреса получателя и отправителя в полях заголовка IP-пакета не изменяются. Зато в заголовке каждого нового кадра, который переносил пакет от одного маршрутизатора к другому, MAC-адреса отправителя и получателя изменяются на каждом отрезке пути.

Процесс доставки DNS-ответа клиенту cit.mgu.com совершенно аналогичен процессу передачи DNS-запроса, который мы только что так подробно описали. Работая в тесной кооперации, протоколы IP, ARP и Ethernet передают клиенту DNS-ответ через всю составную сеть (рис. 15.10).

FTP-клиент, получив IP-адрес FTP-сервера, посылает ему свое сообщение, используя те же описанные ранее механизмы доставки данных через составную сеть. Однако для читателя будет весьма полезно мысленно воспроизвести этот процесс, обращая особое внимание на значения адресных полей заголовков кадров и заголовка вложенного IP-пакета.

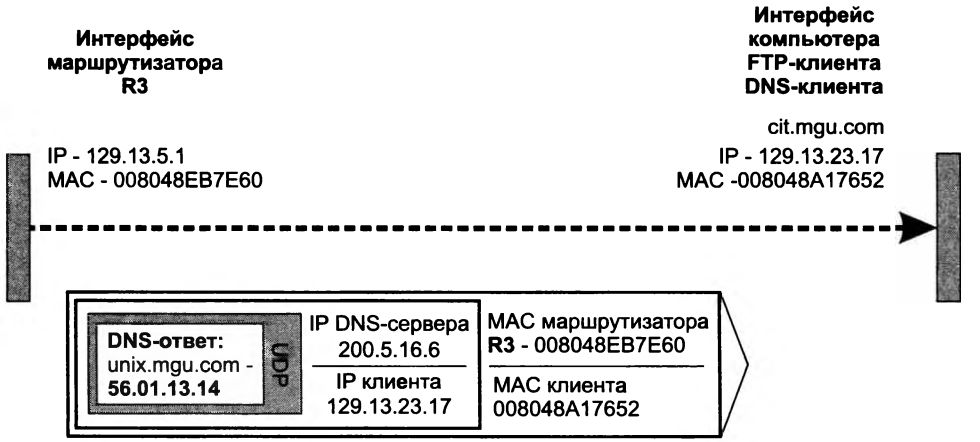


Рис. 15.10. Кадр Ethernet с DNS-ответом, отправленный с маршрутизатора R3 компьютеру-клиенту

Маршрутизация с использованием масок

Алгоритм маршрутизации усложняется, когда в систему адресации узлов вносятся дополнительные элементы — маски. Часто администраторы сетей испытывают неудобства, поскольку количества централизованно выделенных им номеров сетей недостаточно для того, чтобы структурировать сеть надлежащим образом, например развести все слабо взаимодействующие компьютеры по разным сетям. В такой ситуации возможны два пути. Первый из них связан с получением от какого-либо центрального органа дополнительных номеров сетей. Второй способ, употребляющийся чаще, связан с использованием технологии масок, которая позволяет разделить одну имеющуюся сеть на несколько.

Структуризация сети масками одинаковой длины

Допустим, администратор получил в свое распоряжение сеть класса B: 129.44.0.0. Он может организовать сеть с большим числом узлов, номера которых доступны ему из диапазона 0.0.0.1–0.0.255.254. Всего в его распоряжении имеется $(2^{16} - 2)$ адреса (вычитание двойки связано с тем, что, как уже отмечалось, адреса из одних нулей и одних единиц имеют специальное назначение и не годятся для адресации узлов). Однако ему не нужна одна большая неструктурированная сеть. Производственная необходимость диктует администратору другое решение, в соответствии с которым сеть должна быть разделена на три отдельные подсети, при этом трафик в каждой подсети должен быть надежно локализован. Это позволит легче диагностировать сеть и проводить в каждой из подсетей особую политику безопасности. (Заметим, что разделение большой сети с помощью масок имеет еще одно преимущество — оно позволяет скрыть внутреннюю структуру сети предприятия от внешнего наблюдения и тем самым повысить ее безопасность.)

На рис. 15.11 показано разделение всего полученного администратором адресного диапазона на четыре равные части — каждая по 2^{14} адресов. При этом число разрядов, доступное

для нумерации узлов, *уменьшилось* на два бита, а префикс (номер) каждой из четырех сетей стал *длиннее* на два бита. Следовательно, каждый из четырех диапазонов можно записать в виде IP-адреса с маской, состоящей из 18 единиц, или в десятичной нотации — 255.255.192.0.

129.44.0.0/18 (10000001 00101100 **00**000000 00000000)

129.44.64.0/18 (10000001 00101100 **01**000000 00000000)

129.44.128.0/18 (10000001 00101100 **10**000000 00000000)

129.44.192.0/18 (10000001 00101100 **11**000000 00000000)

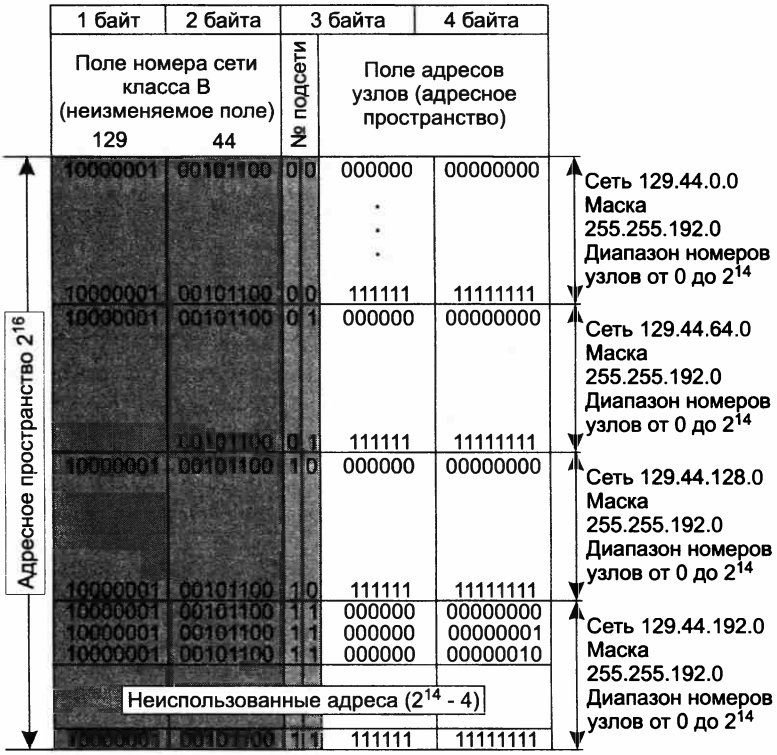


Рис. 15.11. Разделение адресного пространства 129.44.0.0 сети класса B на четыре равные части

Из приведенных записей видно, что администратор получает возможность использовать для нумерации подсетей два дополнительных бита (выделены жирным шрифтом). Именно это позволяет ему сделать из одной централизованно выделенной сети четыре, в данном примере это сети 129.44.0.0/18, 129.44.64.0/18, 129.44.128.0/18, 129.44.192.0/18.

ПРИМЕЧАНИЕ

Некоторые программные и аппаратные маршрутизаторы, следуя устаревшим рекомендациям RFC 950, не поддерживают номера подсетей, которые состоят либо только из одних нулей, либо только из одних единиц. Например, для такого типа оборудования номер сети 129.44.0.0 с маской 255.255.192.0, использованной в нашем примере, окажется недопустимым, поскольку в этом случае

разряды в поле номера подсети имеют значение 00. По аналогичным соображениям недопустимым может оказаться номер сети 129.44.192.0 с тем же значением маски. Здесь номер подсети состоит только из единиц. Однако современные маршрутизаторы, поддерживающие рекомендации RFC 1878, свободны от этих ограничений.

Пример сети, построенной путем деления на четыре сети равного размера, показан на рис. 15.12. Весь трафик во внутреннюю сеть 129.44.0.0, направляемый из внешней сети, поступает через маршрутизатор R1. В целях структуризации информационных потоков во внутренней сети установлен дополнительный маршрутизатор R2. Каждая из вновь образованных сетей 129.44.0.0/18, 129.44.64.0/18, 129.44.128.0/18 и 129.44.192.0/18 подключена к соответственно сконфигурированным портам внутреннего маршрутизатора R2.

ПРИМЕЧАНИЕ

В одной из этих сетей (129.44.192.0/18), выделенной для организации соединения между внешним и внутренним маршрутизаторами, для адресации узлов задействованы всего два адреса — 129.44.192.1 (порт маршрутизатора R2) и 129.44.192.2 (порт маршрутизатора R1). Огромное число узлов в этой подсети не используется. Такой пример выбран исключительно в учебных целях, чтобы показать неэффективность сетей равного размера.

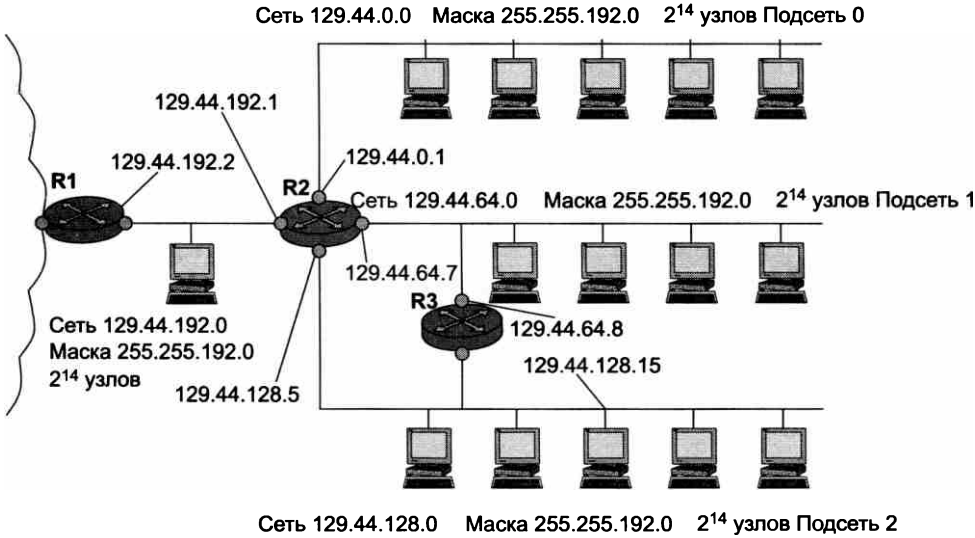


Рис. 15.12. Маршрутизация с использованием масок одинаковой длины

Извне сеть по-прежнему выглядит как единая сеть класса B. Однако поступающий в сеть общий трафик разделяется локальным маршрутизатором R2 между четырьмя сетями. В условиях, когда механизм классов не действует, маршрутизатор должен иметь другое средство, которое позволило бы ему определять, какая часть 32-битного числа, помещенного в поле адрес назначения, является номером сети. Именно этой цели служит дополнительное поле маски, включенное в таблицу маршрутизации (табл. 15.8).

Таблица 15.8. Таблица маршрутизатора R2 в сети с масками одинаковой длины

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.192.0	129.44.0.1	129.44.0.1	Подключена
129.44.64.0	255.255.192.0	129.44.64.7	129.44.64.7	Подключена
129.44.128.0	255.255.192.0	129.44.128.5	129.44.128.5	Подключена
129.44.192.0	255.255.192.0	129.44.192.1	129.44.192.1	Подключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–
129.44.128.15	255.255.255.255	129.44.64.8	129.44.64.7	–

Первые четыре записи в таблице соответствуют внутренним подсетям, непосредственно подключенным к портам маршрутизатора R2.

Запись 0.0.0.0 с маской 0.0.0.0 соответствует маршруту по умолчанию.

Последняя запись определяет специфический маршрут к узлу 129.44.128.15. В тех строках таблицы, в которых в качестве адреса назначения указан полный IP-адрес узла, маска имеет значение 255.255.255.255. В отличие от всех других узлов сети 129.44.128.0, к которым пакеты поступают с интерфейса 129.44.128.5 маршрутизатора R2, к данному узлу они должны приходить через маршрутизатор R3.

Просмотр таблиц маршрутизации с учетом масок

Алгоритм просмотра таблиц маршрутизации, содержащих маски, имеет много общего с описанным алгоритмом просмотра таблиц, не содержащих маски. Однако в нем имеются и существенные изменения.

1. Поиск следующего маршрутизатора для вновь поступившего IP-пакета протокол начинает с того, что *извлекает из пакета адрес назначения* (обозначим его IP_D). Затем протокол IP приступает к процедуре просмотра таблицы маршрутизации, также состоящей из двух фаз, как и процедура просмотра таблицы, в которой столбец маски отсутствует.
2. *Первая фаза* состоит в *поиске специфического маршрута* для адреса IP_D. С этой целью из каждой записи таблицы, в которой маска имеет значение 255.255.255.255, извлекается адрес назначения и сравнивается с адресом из пакета IP_D. Если в какой-либо строке совпадение произошло, то адрес следующего маршрутизатора для данного пакета берется из данной строки.
3. *Вторая фаза* выполняется только в том случае, если во время первой фазы не произошло совпадения адресов. Она состоит в *поиске неспецифического маршрута*, общего для группы узлов, к которой относится и пакет с адресом IP_D. Для этого средствами IP заново просматривается таблица маршрутизации, причем с *каждой* записью производятся следующие действия:
   1) маска (обозначим ее M), содержащаяся в данной записи, «накладывается» на IP-адрес узла назначения IP_D, извлеченный из пакета: IP_D AND M;
   2) полученное в результате число сравнивается со значением, которое помещено в поле адреса назначения той же записи таблицы маршрутизации;

3) если происходит совпадение, протокол IP соответствующим образом *отмечает эту строку*;

4) если просмотрены не все строки, то протокол IP аналогичным образом просматривает следующую строку, если все (включая строку о маршруте по умолчанию), то просмотр записей заканчивается и происходит переход к следующему шагу.

4. После просмотра всей таблицы маршрутизатор выполняет одно их трех действий:
   - если не произошло ни одного совпадения и маршрут по умолчанию отсутствует, то пакет отбрасывается;
   - если произошло одно совпадение, то пакет отправляется по маршруту, указанному в строке с совпавшим адресом;
   - если произошло несколько совпадений, то все помеченные строки сравниваются и выбирается маршрут из той строки, в которой количество совпавших двоичных разрядов наибольшее (другими словами, в ситуации, когда адрес назначения пакета принадлежит сразу нескольким подсетям, маршрутизатор использует наиболее специфический маршрут).

ПРИМЕЧАНИЕ

Во многих таблицах маршрутизации запись с адресом 0.0.0.0 и маской 0.0.0.0 соответствует маршруту по умолчанию. Действительно, любой адрес в пришедшем пакете после наложения на него маски 0.0.0.0 даст адрес сети 0.0.0.0, что совпадает с адресом, указанным в записи. Поскольку маска 0.0.0.0 имеет нулевую длину, то этот маршрут считается самым неспецифическим и используется только при отсутствии совпадений с остальными записями из таблицы маршрутизации.

Проиллюстрируем, как маршрутизатор R2 (см. рис. 15.12) использует описанный алгоритм для работы со своей таблицей маршрутизации (см. табл. 15.8). Пусть на маршрутизатор R2 поступает пакет с адресом назначения 129.44.78.200. Модуль IP, установленный на этом маршрутизаторе, прежде всего сравнит этот адрес с адресом 129.44.128.15, для которого определен специфический маршрут. Совпадения нет, поэтому модуль IP начинает последовательно обрабатывать все строки таблицы, накладывая маски и сравнивая результаты до тех пор, пока не найдет совпадения номера сети в адресе назначения и в строке таблицы. В результате определяется маршрут для пакета 129.44.78.200 — он должен быть отправлен на выходной порт маршрутизатора 129.44.64.7 в сеть 129.44.64.0, непосредственно подключенную к данному маршрутизатору.

Использование масок переменной длины

Во многих случаях более эффективным является разбиение сети на подсети разного размера. В частности, для подсети, которая связывает два маршрутизатора по двухточечной схеме, даже количество адресов сети класса C явно является избыточным.

На рис. 15.13 приведен другой пример распределения того же адресного пространства 129.44.0.0/16, что и в предыдущем примере. Здесь половина из имеющихся адресов (2^{15}) отведена для создания *сети 1*, имеющей адрес 129.44.0.0 и маску 255.255.128.0.

Следующая порция адресов, занимающая адресное пространство 129.44.128.0, что составляет четверть всего адресного пространства (2^{14}), назначена для *сети 2* с маской 255.255.192.0.

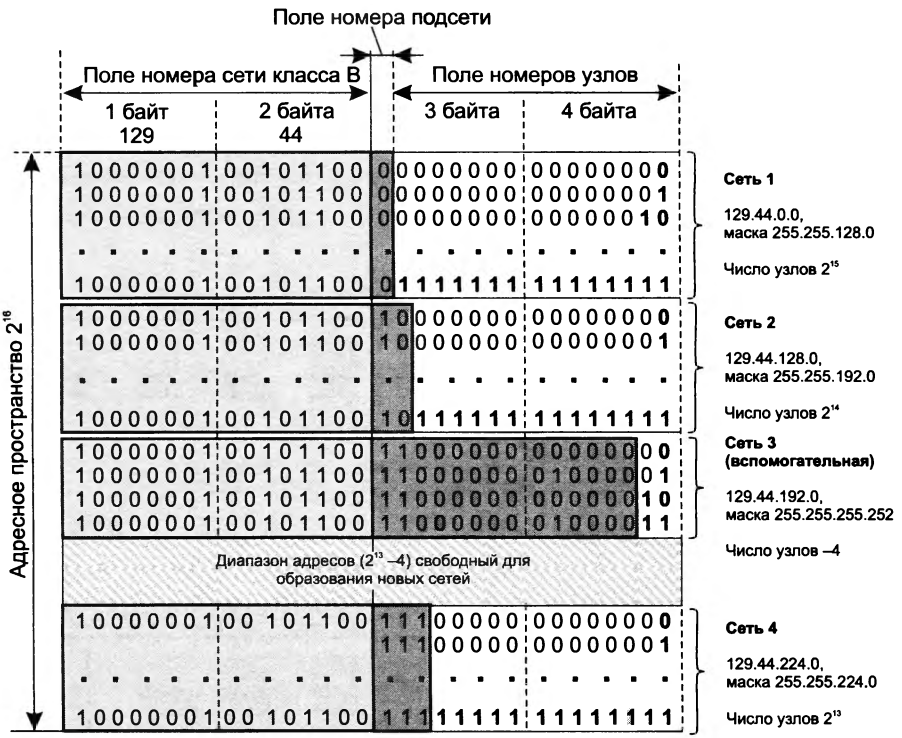


Рис. 15.13. Разделение адресного пространства 129.44.0.0 сети класса B на сети разного размера путем использования масок переменной длины

Далее в пространстве адресов был «вырезан» небольшой фрагмент для создания вспомогательной *сети 3*, предназначенной для связывания внутреннего маршрутизатора R2 с внешним маршрутизатором R1. Для нумерации узлов в такой вырожденной сети достаточно отвести два двоичных разряда. Из четырех возможных комбинаций номеров узлов: 00, 01, 10 и 11 два номера имеют специальное назначение и не могут быть присвоены узлам, но оставшиеся два 10 и 01 позволяют адресовать порты маршрутизаторов. Поле номера узла в таком случае имеет два двоичных разряда, маска в десятичной нотации выглядит так: 255.255.255.252, а номер сети, как видно из рисунка, равен 129.44.192.0.

ПРИМЕЧАНИЕ

Глобальным связям между маршрутизаторами, соединенными по двухточечной схеме, не обязательно давать IP-адреса. Такой интерфейс маршрутизатора называется ненумерованным (unnumbered). Однако чаще всего подобной вырожденной сети все же дают IP-адрес. Помимо прочего, это делается, например, для того чтобы скрыть внутреннюю структуру сети и обращаться к ней по одному адресу входного порта маршрутизатора, в данном примере — по адресу 129.44.192.1, применяя технику трансляции сетевых адресов (Network Address Translation, NAT).

Оставшееся адресное пространство администратор может «нарезать» на разное количество сетей разного объема в зависимости от своих потребностей. Из оставшегося пула (2^{14} – 4) адресов администратор, например, может образовать еще одну достаточно большую сеть

с числом узлов 2^{13} — на рисунке это *сеть 4*. При этом свободными останутся почти столько же адресов ($2^{13} - 4$), которые также могут быть использованы для создания новых сетей. К примеру, из этого «остатка» можно образовать 31 сеть, каждая из которых равна размеру сети класса C, плюс еще несколько сетей меньшего размера. Ясно, что разбиение может быть другим, но, в любом случае, с помощью масок переменного размера администратор имеет возможность более рационально распорядиться всеми имеющимися у него адресами.

На рис. 15.14 показан пример сети, структурированной с помощью масок переменной длины.

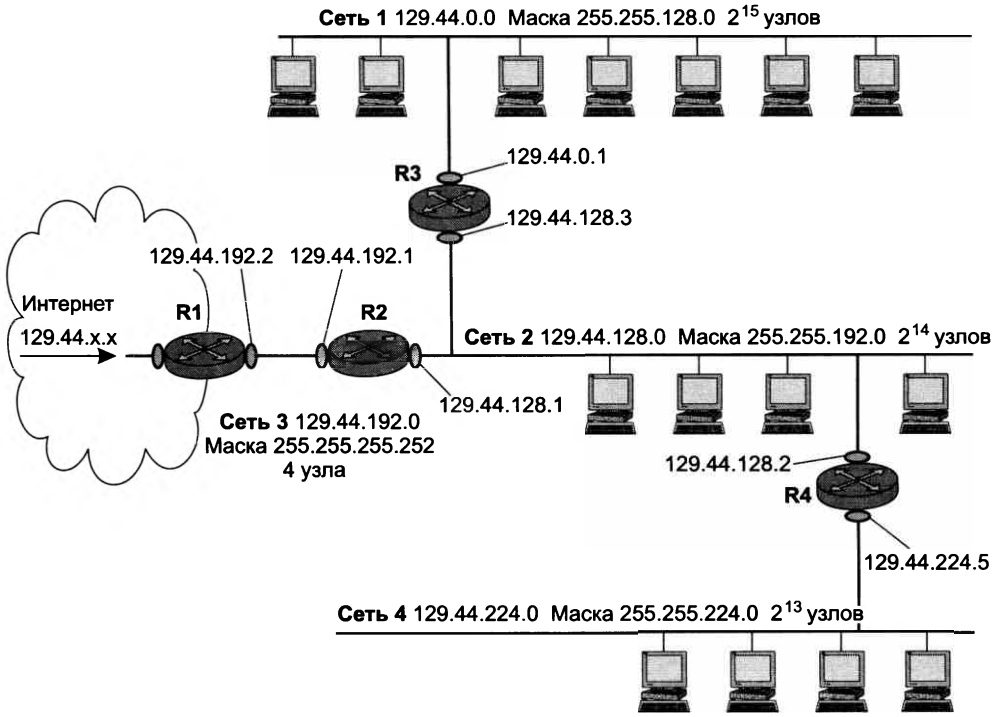


Рис. 15.14. Структуризация сети масками переменной длины

Давайте посмотрим, как маршрутизатор R2 обрабатывает поступающие на его интерфейсы пакеты (табл. 15.9).

Таблица 15.9. Таблица маршрутизатора R2 в сети с масками переменной длины

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.128.0	129.44.128.3	129.44.128.1	1
129.44.128.0	255.255.192.0	129.44.128.1	129.44.128.1	Подключена
129.44.192.0	255.255.255.248	129.44.192.1	129.44.192.1	Подключена
129.44.224.0	255.255.224.0	129.44.128.2	129.44.128.1	1
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	—

Пусть поступивший на R2 пакет имеет адрес назначения 129.44.192.15. Поскольку специфические маршруты в таблице отсутствуют, маршрутизатор переходит ко второй фазе — фазе последовательного анализа строк на предмет поиска совпадения с адресом назначения:

(129.44.192.5) AND (255.255.128.0) = 129.44.128.0 — нет совпадения;

(129.44.192.5) AND (255.255.192.0) = 129.44.192.0 — нет совпадения;

(129.44.192.5) AND (255.255.255.248) = 129.44.192.0 — совпадение;

(129.44.192.5) AND (255.255.224.0) = 129.44.192.0 — нет совпадения.

Таким образом, совпадение имеет место в одной строке. Пакет будет отправлен в непосредственно подключенную к данному маршрутизатору сеть на выходной интерфейс 129.44.192.1.

Если пакет с адресом 129.44.192.1 поступает из внешней сети и маршрутизатор R1 не использует маски, пакет передается маршрутизатору R2, а потом снова возвращается в соединительную сеть. Очевидно, что такие передачи пакета не выглядят рациональными.

Маршрутизация будет более эффективной, если в таблице маршрутизации маршрутизатора R1 задать маршруты масками переменной длины (табл. 15.10). Первая из приведенных двух записей говорит о том, что все пакеты, адреса которых начинаются с префикса 129.44, должны быть переданы на маршрутизатор R2. Эта запись выполняет *агрегирование* адресов всех подсетей, созданных на базе одной сети 129.44.0.0. Вторая строка говорит о том, что среди всех возможных подсетей сети 129.44.0.0 есть одна (129.44.192.0/30), которой пакеты можно направлять непосредственно, а не через маршрутизатор R2.

Таблица 15.10. Фрагмент таблицы маршрутизатора R1

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.0.0	129.44.192.1	129.44.191.2	2
129.44.192.0	255.255.255.192	129.44.192.2	129.44.192.2	Подключена

ПРИМЕЧАНИЕ

При использовании механизма маршрутизации на основе масок в IP-пакетах передается только IP-адрес назначения, а маска сети назначения не передается. Поэтому из IP-адреса пришедшего пакета невозможно выяснить, какая часть адреса относится к номеру сети, а какая — к номеру узла. Если маски во всех подсетях имеют один размер, то это не создает проблем. Если же для образования подсетей применяют маски переменной длины, то маршрутизатор должен как-то узнавать, каким адресам сетей какие маски соответствуют. Для этого используются протоколы маршрутизации, переносящие между маршрутизаторами не только служебную информацию об адресах сетей, но и о масках, соответствующих этим номерам. К таким протоколам относятся протоколы RIPv2 и OSPF, а вот, например, протокол RIP маски не переносит и для маршрутизации на основе масок переменной длины не подходит.

Перекрытие адресных пространств

Со сложностями использования масок администратор впервые сталкивается не тогда, когда начинает конфигурировать сетевые интерфейсы и создавать таблицы маршрутизации, а гораздо раньше — на этапе планирования сети. Планирование включает определение количества сетей, из которых будет состоять корпоративная сеть, оценку требуемого количества адресов для каждой сети, получение пула адресов от поставщика услуг, распределение адресного пространства между сетями. Последняя задача часто оказывается нетривиальной, особенно когда решается в условиях дефицита адресов.

Рассмотрим пример использования масок для организации *перекрывающихся адресных пространств*.

Пусть на некотором предприятии было принято решение обратиться к поставщику услуг для получения пула адресов, достаточного для создания сети, структура которой показана на рис. 15.15. Сеть клиента включает три подсети. Две из них — это надежно защищенные от внешних атак внутренние сети отделов: сеть Ethernet на 600 пользователей и сеть Token Ring на 200 пользователей. Предприятие также предусматривает отдельную открытую для доступа извне сеть на 10 узлов, главное назначение которой — предоставление информации потенциальным заказчикам в режиме открытого доступа. Такого рода фрагменты корпоративной сети, в которых располагаются веб-серверы, FTP-серверы и другие источники публичной информации, называют **демилитаризованной зоной** (Demilitarized Zone, DMZ). Еще одна сеть на 2 узла потребуется для связи с поставщиком услуг, то есть общее число адресов, требуемых для адресации сетевых интерфейсов, составляет 812. Кроме того, необходимо, чтобы пул доступных адресов включал для каждой из сетей широковещательные адреса, состоящие только из единиц, а также адреса, состоящие только из нулей. Учитывая также, что в любой сети адреса всех узлов должны иметь одинаковые префиксы, становится очевидным, что минимальное количество адресов, необходимое клиенту для построения задуманной сети, может значительно отличаться от значения 812, полученного простым суммированием.

В данном примере поставщик услуг решает выделить клиенту непрерывный пул из 1024 адресов. Значение 1024 выбрано как наиболее близкое к требуемому количеству адресов, равному степени двойки ($2^{10} = 1024$). Поставщик услуг выполняет поиск области такого размера в имеющемся у него адресном пространстве — 131.57.0.0/16, часть которого, как показано на рис. 15.16, уже распределена. Обозначим выделенные участки и владеющих ими клиентов через S1, S2 и S3. Поставщик услуг находит среди еще не выделенных адресов непрерывный участок размером 1024 адреса, начальный адрес которого кратен размеру данного участка. Таким образом, наш клиент получает пул адресов 131.57.8.0/22, обозначенный на рисунке через S.

Далее начинается самый сложный этап — распределение полученного от поставщика услуг адресного пула S между четырьмя сетями клиента. Прежде всего администратор решил назначить для самой большой сети (Ethernet на 600 узлов) весь пул адресов 131.57.8.0/22, полученный от поставщика услуг (рис. 15.17). Номер, назначенный для этой сети, совпадает с номером сети, полученным от поставщика услуг. А как же быть с оставшимися тремя сетями? Администратор учел, что для сети Ethernet требуется только 600 адресов, а из оставшихся 624 «выкроил» сеть Token Ring 131.57.9.0/24 на 250 адресов. Воспользовавшись тем, что для Token Ring требуется только 200 адресов, он «вырезал» из нее два участка: для сети DMZ 131.57.9.16/28 на 16 адресов и для связывающей сети 131.57.9.32/30 на четыре адреса. В результате все сети клиента получили достаточное (а иногда и с избытком) количество адресов.

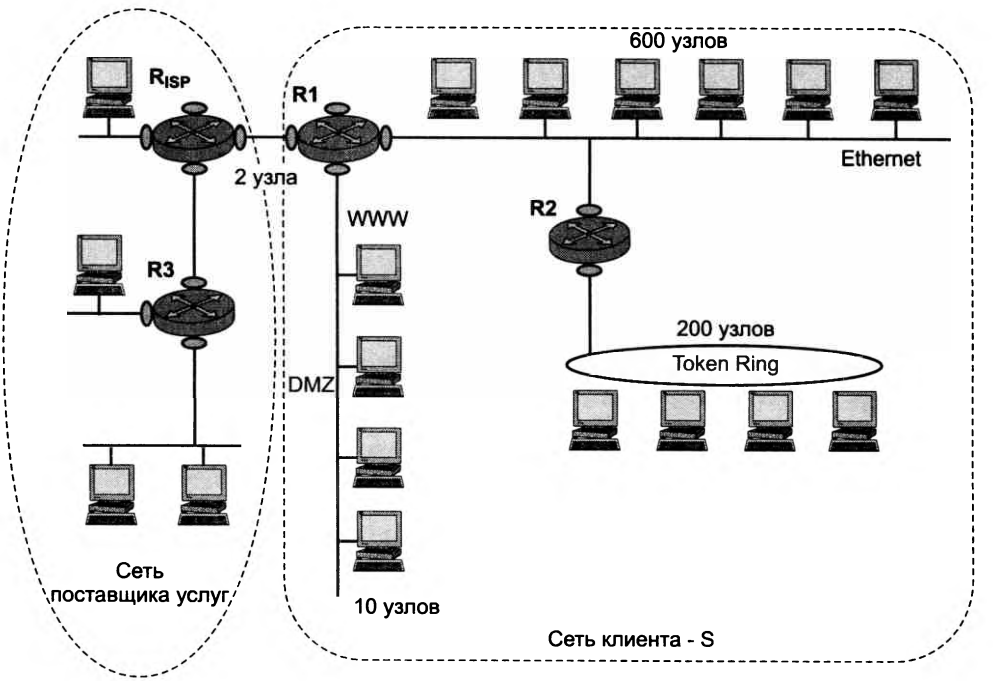


Рис. 15.15. Сети поставщика услуг и клиента

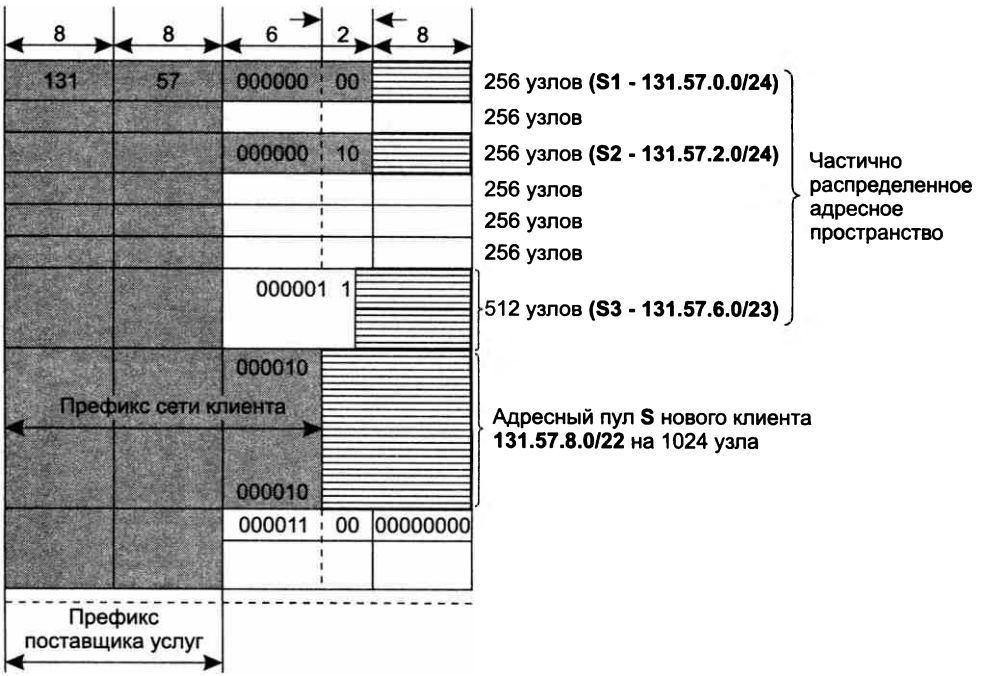


Рис. 15.16. Адресное пространство поставщика услуг

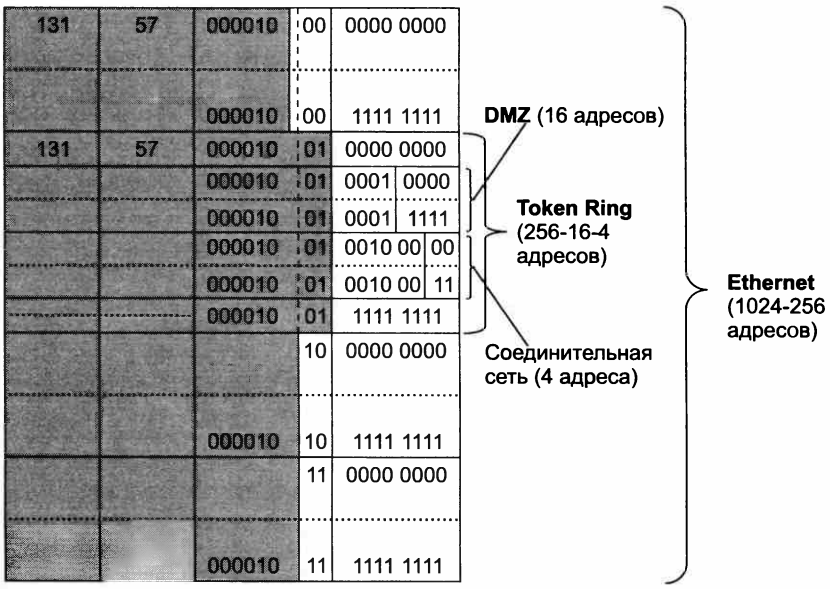


Рис. 15.17. Планирование адресного пространства для сетей клиента

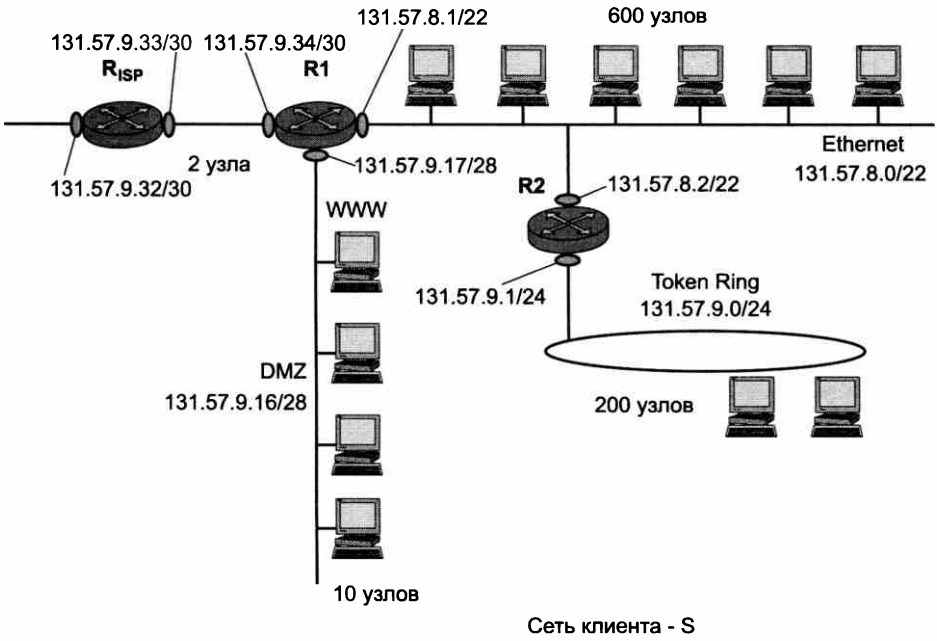


Рис. 15.18. Сконфигурированная сеть клиента

Следующий этап — конфигурирование сетевых интерфейсов конечных узлов и маршрутизаторов. Каждому интерфейсу сообщаются его IP-адрес и соответствующая маска. На рис. 15.18 показана сконфигурированная сеть клиента.

После конфигурирования сетевых интерфейсов должны быть созданы таблицы маршрутизации для маршрутизаторов R1 и R2 клиента. Они могут быть сгенерированы автоматически или с участием администратора. Таблица маршрутизации маршрутизатора R2 соответствует табл. 15.11.

Таблица 15.11. Таблица маршрутизации маршрутизатора R2

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние
131.57.8.0	255.255.252.0	131.57.8.2	131.57.8.2	Подключена
131.57.9.0	255.255.255.0	131.57.9.1	131.57.9.1	Подключена
131.57.9.16	255.255.255.240	131.57.8.1	131.57.8.2	1
131.57.9.32	255.255.255.252	131.57.8.1	131.57.8.2	1

В данной таблице нет маршрута по умолчанию, а значит, все пакеты, адресованные сетям, адреса которых явно не указаны в таблице, будут отбрасываться маршрутизатором.

Пусть, например, на маршрутизатор R2 поступает пакет с адресом назначения 131.57.9.29. В результате просмотра таблицы получаем следующие результаты для каждой строки:

(131.57.9.29) AND (255.255.252.0) = 131.57.8.0 — совпадение;

(131.57.9.29) AND (255.255.255.0) = 131.57.9.0 — совпадение;

(131.57.9.29) AND (255.255.255.240) = 131.57.9.16 — совпадение;

(131.57.9.29) AND (255.255.255.252) = 131.57.9.28 — нет совпадения.

Поскольку при наличии нескольких совпадений выбирается маршрут из той строки, в которой совпадение адреса назначения с адресом из пакета имеет наибольшую длину, определено, что пакет с адресом 131.57.9.29 направляется в сеть DMZ.

CIDR и маршрутизация

За последние годы в Интернете многое изменилось: резко возросло число узлов и сетей, повысилась интенсивность трафика, изменился характер передаваемых данных. Поскольку сегодня таблицы магистральных маршрутизаторов в Интернете могут содержать до нескольких сотен и даже тысяч маршрутов, то из-за перегрузок, вызванных обработкой больших объемов служебной информации, и несовершенства протоколов маршрутизации обмен сообщениями об обновлении таблиц стал приводить к сбоям магистральных маршрутизаторов.

На решение этой проблемы направлена, в частности, технология **бесклассовой междоменной маршрутизации** CIDR. Мы уже говорили в предыдущей главе о том, как CIDR способствует гибкости распределения адресов между владельцами сетей.

Сейчас мы покажем, как эта технология повышает эффективность маршрутизации.

Суть заключается в следующем. Каждому поставщику услуг Интернета назначается *непрерывный* диапазон IP-адресов. При таком подходе *все* адреса каждого поставщика услуг имеют общую старшую часть — **префикс**, поэтому маршрутизация на магистралях Интернета может осуществляться на основе префиксов, а не полных адресов сетей. А это значит, что вместо множества записей по числу сетей в таблицу маршрутизации достаточно поместить *одну запись сразу для всех сетей, имеющих общий префикс*. Такое агрегирование адресов позволит уменьшить объем таблиц в маршрутизаторах всех уровней, а следовательно, ускорить работу маршрутизаторов и повысить пропускную способность Интернета.

Ранее мы рассматривали примеры, где администраторы корпоративных сетей с помощью масок делили на несколько частей непрерывный пул адресов, полученный от поставщика услуг, чтобы использовать эти части для структуризации своей сети. Такой вариант применения масок называется *разделением на подсети*.

Вместе с тем в процессе разделения на подсети с помощью масок проявлялся и обратный эффект их применения. Упрощенно говоря, для того чтобы направить весь суммарный трафик, адресованный из внешнего окружения в корпоративную сеть, разделенную на подсети, достаточно, чтобы во всех внешних маршрутизаторах наличествовала одна строка. В этой строке на месте адреса назначения должен быть указан *общий префикс для всех этих сетей*. Здесь мы имеем дело с операцией, обратной разделению на подсети, — операцией *агрегирования нескольких сетей в одну более крупную*.

Вернемся к рис. 15.16, на котором показано адресное пространство поставщика услуг с участками S1, S2, S3 и S, переданными в пользование четырем клиентам. Этот пример также иллюстрирует рис. 15.19. В результате агрегирования сетей клиентов в табл. 15.12 маршрутизатора R_{ISP} поставщика услуг для каждого клиента будет выделено по одной строке независимо от количества подсетей, организованных ими в своих сетях. Так, вместо четырех маршрутов к четырем сетям клиента S в таблице задан только один общий для всех них маршрут (выделенный жирным шрифтом).

Таблица 15.12. Таблица маршрутизатора RISP поставщика услуг

Адрес назначения	Маска	Следующий маршрутизатор	Номер выходного интерфейса	Расстояние
131.57.0.0 (S1)	255.255.255.0	R3	1	Подключена
131.57.2.0 (S2)	255.255.255.0	R3	3	1
131.57.4.0 (S3)	255.255.254.0	R1	3	1
131.57.8.0 (S)	**255.255.252.0**	**R1**	**2**	**Подключена**
Маршрут по умолчанию	0.0.0.0	$R_{external}$	4	–

Итак, внедрение технологии CIDR позволяет решить две основные задачи.

- *Более экономное расходование адресного пространства.* Благодаря технологии CIDR поставщики услуг получают возможность «нарезать» блоки из выделенного им адресного пространства в точном соответствии с требованиями каждого клиента, при этом у клиента остается пространство для маневра на случай будущего роста.

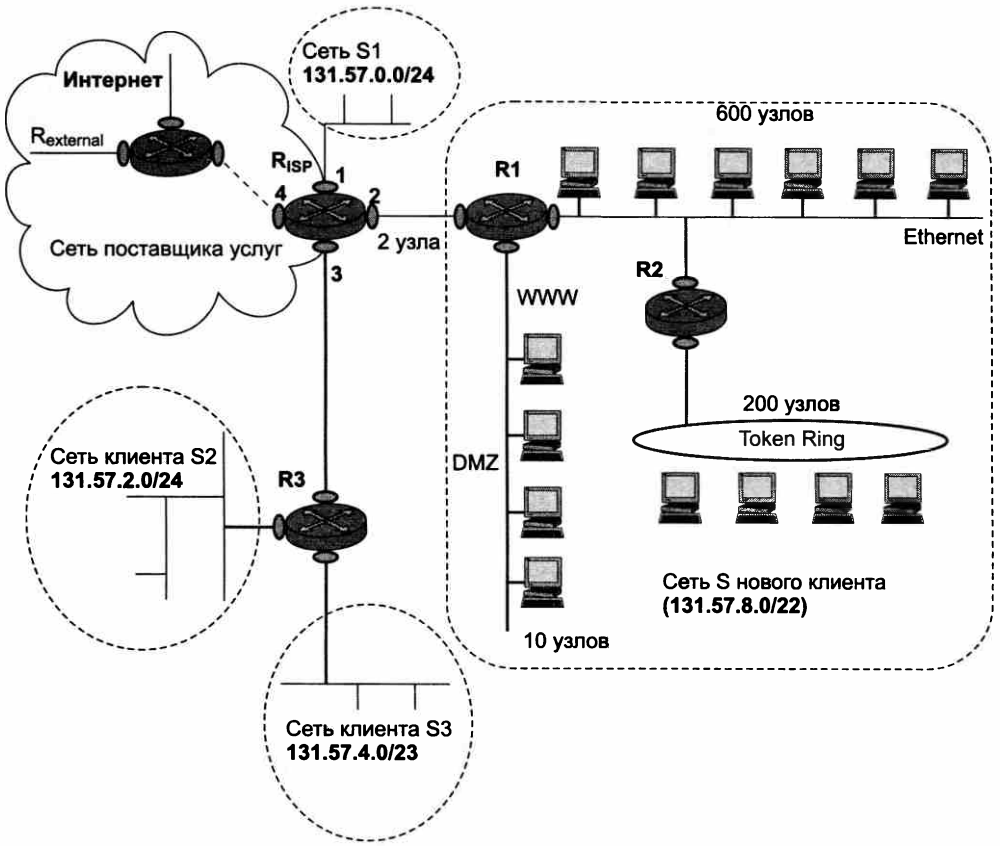


Рис. 15.19. Объединение подсетей

- *Уменьшение числа записей в таблицах маршрутизации* за счет объединения маршрутов — одна запись в таблице маршрутизации может представлять большое количество сетей. Если все поставщики услуг Интернета начнут придерживаться стратегии CIDR, то особенно заметный выигрыш будет достигаться в магистральных маршрутизаторах.

Необходимым условием эффективного использования технологии CIDR является **локализация адресов**, то есть назначение адресов, имеющих совпадающие префиксы, сетям, располагающимся территориально по соседству. Только в таком случае трафик может быть агрегирован.

К сожалению, сейчас распределение адресов носит во многом случайный характер. Кардинальный путь решения проблемы — перенумерование сетей. Однако эта процедура сопряжена с определенными временны́ми и материальными затратами и для ее проведения пользователей нужно каким-либо образом стимулировать. В качестве таких стимулов рассматривается, например, введение оплаты за строку в таблице маршрутизации или же за количество узлов в сети. Первое требование подводит потребителя к мысли получить у поставщика услуг такой адрес, чтобы маршрутизация трафика в его сеть шла на основании префикса и номер его сети не фигурировал больше в магистральных маршрутизаторах.

Требование оплаты каждого адреса узла также может подтолкнуть потребителя решиться на перенумерование, чтобы получить ровно столько адресов, сколько ему нужно.

Технология CIDR уже успешно используется в текущей версии протокола IP (IPv4) и поддерживается такими протоколами маршрутизации, как OSPF, RIP-2, BGP4 (в основном на магистральных маршрутизаторах Интернета). Особенности применения технологии CIDR в новой версии протокола IP (IPv6) рассматриваются в конце этой главы.

Фрагментация IP-пакетов

Важной особенностью протокола IP, отличающей его от других сетевых протоколов (например, от сетевого протокола IPX, который какое-то время назад конкурировал с IP), является его способность выполнять *динамическую фрагментацию* пакетов при передаче их между сетями с различными максимально допустимыми значениями длины поля данных кадров (Maximum Transmission Unit, **MTU**). Значения MTU зависят как от протокола, так и от настройки сетевых интерфейсов.

Прежде всего отметим разницу между фрагментацией сообщений *в узле-отправителе* и динамической фрагментацией сообщений *в транзитных узлах* сети — маршрутизаторах.

В первом случае деление сообщения на несколько более мелких частей (фрагментация) происходит при передаче данных между протоколами одного и того же стека внутри компьютера. Протоколы, выполняющие фрагментацию в пределах узла, анализируют тип технологии нижнего уровня, определяют ее MTU и делят сообщения на такие части, которые умещаются в кадры канального уровня того же стека протоколов.

В стеке TCP/IP эту задачу решает протокол TCP, который разбивает поток байтов, передаваемых ему с прикладного уровня, на сегменты нужного размера, например по 1460 байт, если на нижнем уровне данной сети работает протокол Ethernet. Протокол IP в узле-отправителе, как правило, не использует свои возможности по фрагментации пакетов.

А вот на транзитном узле — маршрутизаторе, когда пакет необходимо передать из сети с большим значением MTU в сеть с меньшим значением MTU, способности протокола IP выполнять фрагментацию становятся востребованными. *Пакеты-фрагменты*, путешествуя по сети, могут вторично подвергнуться фрагментации на каком-либо из промежуточных маршрутизаторов.

Параметры фрагментации

Каждый из фрагментов должен быть снабжен полноценным заголовком IP. Некоторые из полей заголовка (идентификатор, TTL, флаги DF и MF, смещение) непосредственно предназначены для последующей *сборки* фрагментов в исходное сообщение.

- **Идентификатор** пакета используется для *распознавания* пакетов, образовавшихся путем деления на части (фрагментации) исходного пакета. Все части (фрагменты) одного пакета должны иметь одинаковое значение этого поля. Модуль IP, отправляющий пакет, устанавливает в поле идентификатора значение, которое должно быть уникальным для данной пары отправителя и получателя в течение всего времени, пока данный пакет (или любой его фрагмент) может существовать в составной IP-сети.
- Поле **времени жизни** (Time To Live, TTL) занимает один байт и определяет предельный срок, в течение которого пакет может перемещаться по сети. Время жизни пакета изме-

ряется в секундах и задается источником (отправителем). Как уже отмечалось в начале этой главы, по истечении каждой секунды пребывания на каждом из маршрутизаторов, через которые проходит пакет во время своего «путешествия» по сети, из его текущего времени жизни вычитается единица; единица вычитается и в том случае, если время пребывания было меньше секунды. Поскольку современные маршрутизаторы редко обрабатывают пакет дольше чем за одну секунду, то время жизни можно интерпретировать как максимальное число транзитных узлов, которые разрешено пройти пакету. Если значение поля времени жизни становится нулевым до того, как пакет достигает получателя, пакет уничтожается. При сборке фрагментов хост-получатель использует значение TTL как крайний срок ожидания недостающих фрагментов.

- Поле **смещения фрагмента** предоставляет получателю информацию о положении фрагмента относительно начала поля данных исходного нефрагментированного пакета. Так, например, первый фрагмент будет иметь в поле смещения нулевое значение. В пакете, не разбитом на фрагменты, поле смещения также имеет нулевое значение. Смещение задается в байтах и должно быть кратно 8 байт.
- Установленный в единицу однобитный флаг **MF** (More Fragments — больше фрагментов) говорит о том, что данный пакет является промежуточным (не последним) фрагментом. Модуль IP, отправляющий нефрагментированный пакет, устанавливает бит MF в нуль.
- Флаг **DF** (Do not Fragment — не фрагментировать), установленный в единицу, запрещает маршрутизатору фрагментировать данный пакет. Если помеченный таким образом пакет не может достичь получателя без фрагментации, то модуль IP его уничтожает, а узлу-отправителю посылается диагностическое сообщение.

Возможность запретить фрагментацию позволяет в некоторых случаях ускорить работу приложений. Для этого вначале необходимо исследовать сеть, определить максимальный размер пакета, который сможет пройти весь путь без фрагментации, а затем использовать пакеты такого или меньшего размера для обмена данными. Данная возможность позволяет также предотвратить фрагментацию в тех случаях, когда хост-получатель не имеет достаточных ресурсов для сборки фрагментов.

Механизм фрагментации

Рассмотрим механизм фрагментации на примере составной сети, показанной на рис. 15.20.

В одной из подсетей (Frame Relay) значение MTU равно 4080, в другой (Ethernet) — 1492. Хост, принадлежащий сети Frame Relay, передает данные хосту в сети Ethernet. На обоих хостах, а также на маршрутизаторе, связывающем эти подсети, установлен стек протоколов TCP/IP.

Транспортному уровню *хоста-отправителя* известно значение MTU нижележащей технологии (4080). На основании этого модуль TCP «нарезает» свои сегменты размером 4000 байт и передает вниз протоколу IP, который помещает сегменты в поле данных IP-пакетов и генерирует для них заголовки. Обратим особое внимание на заполнение тех полей заголовка, которые прямо связаны с фрагментацией:

- пакету присваивается уникальный *идентификатор*, например 12456;
- поскольку пакет пока еще не фрагментирован, в поле *смещения* помещается значение 0;

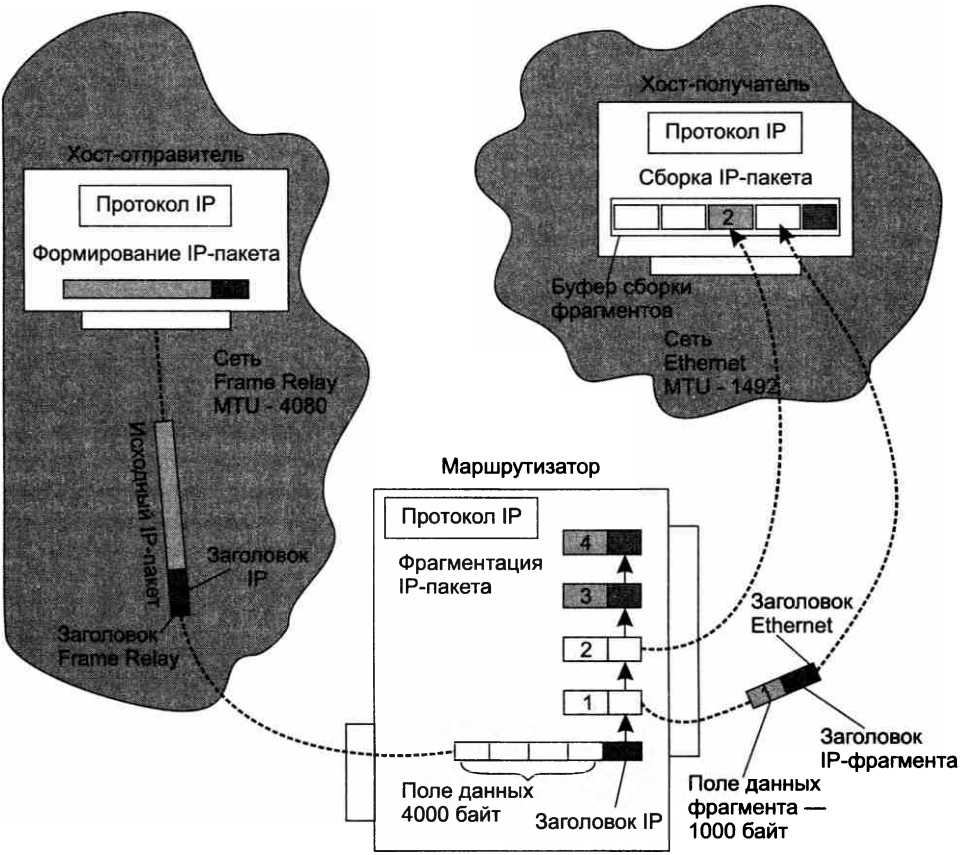


Рис. 15.20. Фрагментация в составной сети

- признак *MF* также обнуляется, это показывает, что пакет одновременно является и своим последним фрагментом;
- признак *DF* устанавливается в 1, это означает, что данный пакет можно фрагментировать.

Общая величина IP-пакета составляет 4000 плюс 20 (размер заголовка IP), то есть 4020 байт, что умещается в поле данных кадра Frame Relay, которое в данном примере равно 4080. Далее модуль IP хоста-отправителя передает этот кадр своему сетевому интерфейсу Frame Relay, который отправляет кадры следующему маршрутизатору.

Модуль IP следующего маршрутизатора по сетевому адресу прибывшего IP-пакета определяет, что пакет нужно передать в сеть Ethernet. Однако она имеет значение MTU, равное 1492, что значительно меньше размера поступившего на входной интерфейс пакета. Следовательно, IP-пакет необходимо фрагментировать. Модуль IP выбирает размер поля данных фрагмента равным 1000, так что из одного большого IP-пакета получается четыре маленьких пакета-фрагмента. Для каждого фрагмента и его заголовка IP в маршрутизаторе создается отдельный буфер (на рисунке фрагменты и соответствующие им буферы

пронумерованы от 1 до 4). Протокол IP копирует в эти буферы содержимое некоторых полей заголовка IP исходного пакета, создавая тем самым «заготовки» заголовков IP всех новых пакетов-фрагментов. Одни параметры заголовка IP копируются в заголовки всех фрагментов, другие — лишь в заголовок первого фрагмента.

В процессе фрагментации могут измениться значения некоторых полей заголовков IP в пакетах-фрагментах по сравнению с заголовком IP исходного пакета. Так, каждый фрагмент имеет собственные значения контрольной суммы заголовка, смещения фрагмента и общей длины пакета. Во всех пакетах, кроме последнего, флаг MF устанавливается в единицу, а в последнем фрагменте — в нуль. Полученные пакеты-фрагменты имеют длину 1020 байт (с учетом заголовка IP), поэтому они свободно помещаются в поле данных кадров Ethernet.

На рисунке показаны разные стадии перемещения фрагментов по сети. Фрагмент 2 уже достиг хоста-получателя и помещен в приемный буфер. Фрагмент 1 еще перемещается по сети Ethernet, остальные фрагменты находятся в буферах маршрутизатора.

А теперь обсудим, как происходит *сборка фрагментированного пакета на хосте назначения.*

ПРИМЕЧАНИЕ

Отметим, что IP-маршрутизаторы не собирают фрагменты пакетов в более крупные пакеты, даже если на пути встречается сеть, допускающая такое укрупнение. Это связано с тем, что отдельные фрагменты сообщения могут перемещаться по составной сети разными маршрутами, поэтому нет гарантии, что все фрагменты на своем пути пройдут через какой-то один определенный маршрутизатор.

На хосте назначения для каждого фрагментированного пакета отводится отдельный буфер. В этот буфер принимающий протокол IP помещает IP-фрагменты, у которых совпадают IP-адреса отправителя и получателя, а также значения в полях идентификатора (в нашем примере — 12456). Все эти признаки говорят модулю IP, что данные пакеты являются фрагментами одного исходного пакета. Сборка заключается в помещении данных из каждого фрагмента в позицию, определенную *смещением*, указанным в заголовке фрагмента.

Когда первый фрагмент исходного пакета приходит на хост-получатель, этот хост запускает *таймер*, который определяет максимальное время ожидания прибытия остальных фрагментов данного пакета. В различных реализациях IP применяются разные правила выбора максимального времени ожидания. В частности, таймер может быть установлен на фиксированный период времени (от 60 до 120 секунд), рекомендуемый RFC. Как правило, этот интервал достаточен для доставки пакета от отправителя получателю. В других реализациях максимальное время ожидания определяется с помощью адаптивных алгоритмов измерения и статистической обработки временны́х параметров сети, позволяющих оценивать ожидаемое время прибытия фрагментов. Наконец, тайм-аут может быть выбран на базе значений TTL прибывающих фрагментов. Последний подход основан на том, что нет смысла ожидать, пока прибудут другие фрагменты пакета, если время жизни одного из прибывших фрагментов уже истекло.

Если хотя бы один фрагмент пакета не успеет прийти на хост назначения к моменту истечения таймера, то никаких действий по дублированию отсутствующего фрагмента не предпринимается, а все полученные к этому времени фрагменты пакета отбрасываются! Хосту, пославшему исходный пакет, направляется ICMP-сообщение об ошибке. Такому поведению протокола IP вполне соответствует его кредо «по возможности» — стараться, но никаких гарантий не давать.

Признаками окончания сборки являются отсутствие незаполненных промежутков в поле данных и прибытие последнего фрагмента (с равным нулю флагом MF) до истечения тайм-аута. После того как данные собраны, их можно передать вышележащему протоколу, например TCP.

Протокол ICMP

Протокол межсетевых управляющих сообщений (Internet Control Message Protocol, ICMP) является вспомогательным протоколом, использующимся для диагностики и мониторинга сети.

Можно представить ряд ситуаций, когда протокол IP не может доставить пакет адресату, например истекает время жизни пакета, в таблице маршрутизации отсутствует маршрут к заданному в пакете адресу назначения, пакет не проходит проверку по контрольной сумме, шлюз не имеет достаточно места в своем буфере для передачи какого-либо пакета и т. д. и т. п.

Свойство «необязательности» протокола IP, доставляющего данные «по возможности», компенсируется протоколами более высоких уровней стека TCP/IP, например TCP на транспортном уровне и в какой-то степени DNS на прикладном уровне. Они берут на себя обязанности по обеспечению надежности, применяя такие известные приемы, как нумерация сообщений, подтверждение доставки, повторная посылка данных.

Протокол ICMP также призван компенсировать ненадежность протокола IP, но несколько *иначе*. Он не предназначен для исправления возникших при передаче пакета проблем: если пакет потерян, ICMP не может послать его заново. Задача ICMP другая — он является *средством оповещения* отправителя о «несчастных случаях», произошедших с его пакетами. Пусть, например, протокол IP, работающий на каком-либо маршрутизаторе, обнаружил, что пакет для дальнейшей передачи по маршруту необходимо фрагментировать, но в пакете установлен признак DF (не фрагментировать). Протокол IP, обнаруживший, что не может передать IP-пакет далее по сети, прежде чем отбросить пакет, должен отправить *диагностическое* ICMP-сообщение конечному узлу-источнику. Для передачи по сети ICMP-сообщение инкапсулируется в поле данных IP-пакета. IP-адрес узла-источника определяется из заголовка пакета, вызвавшего инцидент.

Сообщение, прибывшее в узел-источник, может быть обработано там либо ядром операционной системы, либо протоколами транспортного и прикладного уровней, либо приложениями, либо просто проигнорировано. Важно, что обработка ICMP-сообщений не входит в обязанности протоколов IP и ICMP.

Заметим, что некоторые из пакетов могут исчезнуть в сети, не вызвав при этом никаких оповещений. В частности, протокол ICMP не предусматривает передачу сообщений о проблемах, возникающих при обработке IP-пакетов, несущих ICMP-сообщения об ошибках. Такое решение было принято разработчиками протокола, чтобы не порождать «штормы» в сетях, когда количество сообщений об ошибках лавинообразно возрастает.

Особенностью протокола ICMP является функциональное разнообразие решаемых задач, а следовательно, и связанных с этим сообщений. Все типы сообщений имеют один и тот же формат (рис. 15.21), однако интерпретация полей существенно зависит от того, к какому типу относится сообщение.

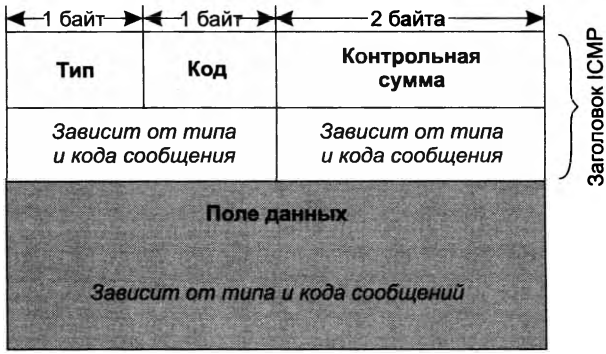


Рис. 15.21. Формат ICMP-сообщения

Заголовок ICMP-сообщения состоит из 8 байт:

- **тип** (1 байт) — числовой идентификатор типа сообщения;
- **код** (1 байт) — числовой идентификатор, более тонко дифференцирующий тип ошибки;
- **контрольная сумма** (2 байта) — подсчитывается для всего ICMP-сообщения.

Содержимое оставшихся четырех байтов в заголовке и поле данных зависят от значений полей типа и кода.

На рис. 15.22 показана таблица основных типов ICMP-сообщений. Эти сообщения можно разделить на две группы (помеченные на рисунке условными символами):

- сообщения об *ошибках*;
- сообщения *запрос-ответ*.

Сообщения типа запрос-ответ связаны в пары: эхо-запрос — эхо-ответ, запрос маски — ответ маски, запрос времени — ответ времени. Отправитель сообщения-запроса всегда рассчитывает на получение соответствующего сообщения-ответа.

Сообщения, относящиеся к группе сообщений об ошибках, конкретизируются уточняющим кодом. На рисунке показан фрагмент таблицы кодов для сообщения об ошибке недостижимости узла назначения, имеющей тип 3. Из таблицы мы видим, что это сообщение может быть вызвано различными причинами, такими как неверный адрес сети или узла (код 0 или 1), отсутствие на конечном узле-адресате необходимого протокола прикладного уровня (код 2 — «протокол недостижим») или открытого порта UDP/TCP (код 3 — «порт недостижим»). Узел (или сеть) назначения может быть также недостижим по причине временной неработоспособности аппаратуры или из-за того, что маршрутизатор не имеет данных о пути к сети назначения. Всего таблица содержит 15 кодов. Аналогичные таблицы кодов существуют и для других типов сообщений об ошибках.

Утилита traceroute

В качестве примера рассмотрим использование сообщений об ошибках в популярной утилите traceroute, предназначнной для мониторинга сети.

Таблица типов ICMP-сообщений

Значение в поле «Тип»	Тип сообщения
0	Эхо-ответ
3	Узел назначения недостижим
4	Подавление источника
5	Перенаправление маршрута
8	Эхо-запрос
11	Истечение времени диаграммы
12	Проблема с параметрами пакета
13	Запрос отметки времени
14	Ответ отметки времени
17	Запрос маски
18	Ответ маски

Таблица кодов причин ошибок 3

Код	Причина
0	Сеть недостижима
1	Узел недостижим
2	Протокол недостижим
3	Порт недостижим
4	Ошибка фрагментации
5	Ошибка в маршруте источника
6	Сеть назначения неизвестна
7	Узел назначения неизвестен
8	Узел-источник изолирован
9	Административный запрет
	. . . . .

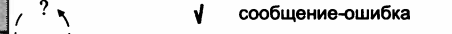


Рис. 15.22. Типы и коды ICMP-сообщений

Когда маршрутизатор не может передать или доставить IP-пакет, он отсылает узлу, отправившему этот пакет, сообщение о недостижимости узла назначения. Формат этого сообщения показан на рис. 15.23. В поле типа помещается значение 3, а в поле кода — значение из диапазона 0–15, уточняющее причину, по которой пакет не был доставлен. Следующие за полем контрольной суммы четыре байта заголовка *не используются* и заполняются нулями.

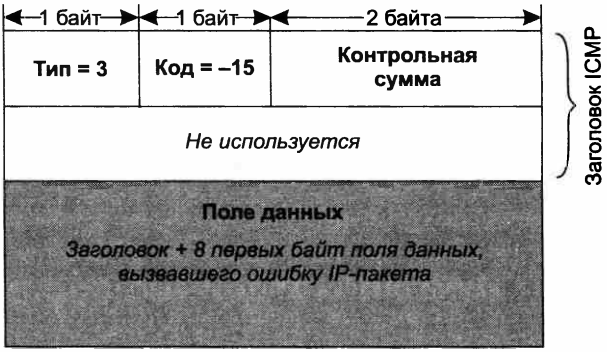


Рис. 15.23. Формат ICMP-сообщения об ошибке недостижимости узла назначения

Помимо причины ошибки, указанной в заголовке (в полях типа и кода), дополнительная диагностическая информация передается в поле данных ICMP-сообщения. Именно туда

помещается *заголовок IP и первые 8 байт данных* того IP-пакета, который вызвал ошибку. Эта информация позволяет узлу-отправителю еще точнее диагностировать причину ошибки. Это возможно, так как все протоколы стека TCP/IP, использующие для передачи своих сообщений IP-пакеты, помещают наиболее важную для анализа информацию в первые 8 байт своих сообщений. В частности, ими вполне могут оказаться первые 8 байт заголовка TCP или UDP, в которых содержится информация (номер порта), идентифицирующая приложение, пославшее потерянный пакет. Следовательно, при разработке приложения можно предусмотреть встроенные средства реакции на сообщения о недоставленных пакетах.

ICMP-сообщения об ошибках лежат в основе работы популярной утилиты traceroute для Unix, имеющей в Windows название tracert. Эта утилита позволяет проследить маршрут до удаленного хоста, определить среднее время оборота (RTT), IP-адрес и в некоторых случаях доменное имя каждого промежуточного маршрутизатора. Такая информация помогает найти маршрутизатор, на котором оборвался путь пакета к удаленному хосту.

Утилита traceroute осуществляет трассировку маршрута, посылая серию обычных IP-пакетов в конечную точку изучаемого маршрута. Идея метода состоит в следующем. Значение времени жизни (TTL) первого отправляемого пакета устанавливается равным 1. Когда протокол IP первого маршрутизатора принимает этот пакет, то он в соответствии со своим алгоритмом уменьшает значение TTL на 1 и получает 0. Маршрутизатор отбрасывает пакет с нулевым временем жизни и возвращает узлу-источнику ICMP-сообщение об ошибке истечения времени дейтаграммы (значение поля типа равно 11) вместе с заголовком IP и первыми 8 байтами потерянного пакета.

Получив ICMP-сообщение о причине недоставки пакета, утилита traceroute запоминает адрес первого маршрутизатора (который извлекает из заголовка IP-пакета, несущего ICMP-сообщение).

Затем traceroute посылает следующий IP-пакет, но теперь со значением TTL, равным 2. Этот пакет благополучно проходит первый маршрутизатор, но «умирает» на втором, о чем немедленно отправляется аналогичное ICMP-сообщение об ошибке истечения времени дейтаграммы. Утилита traceroute запоминает адрес второго маршрутизатора и т. д. Такие действия выполняются с каждым маршрутизатором вдоль маршрута вплоть до узла назначения или неисправного маршрутизатора. Мы рассматриваем работу утилиты traceroute весьма схематично, но и этого достаточно, чтобы оценить изящество идеи, лежащей в основе ее работы. Остальные ICMP-сообщения об ошибках имеют такой же формат и отличаются друг от друга только значениями полей типа и кода.

Далее приведена копия экранной формы, выведенной утилитой tracert (Windows) при трассировке хоста ds.internic.net [198.49.45.29]:

```
1 311 ms 290 ms 261 ms 144.206.192.100
2 281 ms 300 ms 271 ms 194.85.73.5
3 2023 ms 290 ms 311 ms moscow-m9-2-S5.relcom.eu.net [193.124.254.37]
4 290 ms 261 ms 280 ms MSK-M9-13.Relcom.EU.net [193.125.15.13]
5 270 ms 281 ms 290 ms MSK.RAIL-1-ATM0-155Mb.Relcom.EU.net [193.124.254.82]
6 300 ms 311 ms 290 ms SPB-RASCOM-1-E3-1-34Mb.Relcom.EU.net [193.124.254.78]
7 311 ms 300 ms 300 ms Hssi11-0.GW1.STK2.ALTER.NET [146.188.33.125]
8 311 ms 330 ms 291 ms 421.ATM6-0-0.CR2.STK2.Alter.Net [146.188.5.73]
9 360 ms 331 ms 330 ms 219.Hssi4-0.CR2.LND1.Alter.Net [146.188.2.213]
10 351 ms 330 ms 331 ms 412.Atm5-0.BR1.LND1.Alter.net [146.188.3.205]
11 420 ms 461 ms 420 ms 167.ATM8-0-0.CR1.ATL1.Alter.Net [137.39.69.182]12 461 ms 441
```

```
ms 440 ms 311.ATM12-0-0.BR1.ATL1.Alter.Net [137.39.21.73]13 451 ms 410 ms 431 ms
atlanta1-br1.bbnplanet.net [4.0.2.141]14 420 ms 411 ms 410 ms vienna1-br2.bbnplanet.
net [4.0.3.154]15 411 ms 430 ms 2514 ms vienna1-nbr3.bbnplanet.net [4.0.3.150]16
430 ms 421 ms 441 ms vienna1-nbr2.bbnplanet.net [4.0.5.45]17 431 ms 451 ms 420
ms cambridge1-br1.bbnplanet.net [4.0.5.42]18 450 ms 461 ms 441 мс cambridge1-
cr14.bbnplanet.net [4.0.3.94]19 451 мс 461 мс 460 мс attbcstoll.bbnplanet.net
[206.34.99.38]20 501 мс 460 мс 481 мс shutdown.ds.internic.net [198.49.45.29]
```

Последовательность строк соответствует последовательности маршрутизаторов, образующих маршрут к заданному узлу. Первое число в строке — число хопов до соответствующего маршрутизатора. Утилита traceroute тестирует каждый маршрутизатор трижды, поэтому следующие три числа в строке — это значения RTT, вычисленные путем посылки трех пакетов, время жизни которых истекло на этом маршрутизаторе. Если ответ от какого-либо маршрутизатора не приходит за заданное время, то вместо времени на экране печатается звездочка (*).

Далее идут IP-адрес и доменное имя (если оно имеется) маршрутизатора. Видно, что почти все интерфейсы маршрутизаторов поставщиков услуг Интернета зарегистрированы в службе DNS, а первые два, относящиеся к локальным маршрутизаторам, — нет.

Еще раз подчеркнем, что время, указанное в каждой строке, это не время прохождения пакетов между двумя соседними маршрутизаторами, а время, за которое пакет проделывает путь от источника до соответствующего маршрутизатора и обратно. Так как ситуация в Интернете с загрузкой маршрутизаторов постоянно меняется, то время достижимости маршрутизаторов не всегда нарастает монотонно, а может изменяться достаточно произвольным образом.

Утилита ping

А сейчас давайте рассмотрим представителей другой группы ICMP-сообщений — **эхо-запросы** и **эхо-ответы** — и поговорим об использовании этих сообщений в известной утилите ping.

Эхо-запрос и эхо-ответ, в совокупности называемые **эхо-протоколом**, представляют собой очень простое средство мониторинга сети. Компьютер или маршрутизатор посылает по составной сети ICMP-сообщение эхо-запроса, указывая в нем IP-адрес узла, достижимость которого нужно проверить. Узел, получивший эхо-запрос, формирует и отправляет эхо-ответ отправителю запроса. Так как эхо-запрос и эхо-ответ передаются по сети внутри IP-пакетов, то их успешная доставка означает нормальное функционирование всей транспортной системы составной сети.

Формат эхо-запроса и эхо-ответа показан на рис. 15.24. Поле типа для эхо-ответа равно 0, для эхо-запроса — 8; поле кода всегда равно 0 и для запроса, и для ответа. В байтах 5 и 6 заголовка содержится **идентификатор запроса**, в байтах 7 и 8 — **порядковый номер**. В поле данных эхо-запроса может быть помещена произвольная информация, которая в соответствии с данным протоколом должна быть скопирована в поле данных эхо-ответа.

Поля идентификатора запроса и порядкового номера используются одинаковым образом всеми сообщениями типа запрос-ответ. Посылая запрос, приложение помещает в эти два поля информацию, которая предназначена для последующего встраивания ее в соответствующий ответ. Сообщение-ответ копирует значения этих полей в свои поля того же назначения. Когда ответ возвращается в пункт отправки сообщения-запроса,

Рис. 15.24. Формат ICMP-сообщений типа эхо-запрос и эхо-ответ

то на основании идентификатора он может «найти и опознать» приложение, пославшее запрос. А порядковый номер используется приложением, чтобы связать полученный ответ с соответствующим запросом (учитывая, что одно приложение может выдать несколько однотипных запросов).

Утилита ping обычно посылает серию эхо-запросов к тестируемому узлу и предоставляет пользователю статистику об утерянных эхо-ответах и среднем времени реакции сети на запросы. Утилита ping выводит на экран сообщения следующего вида обо всех поступивших ответах:

```
# ping server1.citmgu.ru
Pinging server1.citmgu.ru [193.107.2.200] with 64 bytes of data:
Reply from 193.107.2.200: bytes=64 time=256ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=310ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=260ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=146ms TTL= 123
```

Из приведенной распечатки видно, что в ответ на тестирующие запросы, посланные узлу server1.mgu.ru, было получено четыре эхо-ответа. Длина каждого сообщения составляет 64 байта. В следующей колонке помещены значения времени оборота (RTT), то есть времени от момента отправки запроса до получения ответа на этот запрос. Как видим, сеть работает достаточно нестабильно — время в последней строке отличается от времени во второй более чем в два раза. На экран выводится также оставшееся время жизни поступивших пакетов.

IPv6 как развитие стека TCP/IP

В начале 90-х годов стек протоколов TCP/IP столкнулся с серьезными проблемами. Именно в это время началось активное промышленное использование Интернета: переход к построению сетей предприятий на основе транспорта Интернета, применение веб-технологии для доступа к корпоративной информации, ведение электронной коммерции через Интернет, внедрение Интернета в индустрию развлечений (распространение видеофильмов, звукозаписей, интерактивные игры).

Все это привело к резкому росту числа узлов сети (в начале 90-х годов новый узел в Интернете появлялся каждые 30 секунд), изменению характера трафика и ужесточению требований, предъявляемых к качеству обслуживания сетью ее пользователей.

Сообщество Интернета, а вслед за ним и весь телекоммуникационный мир, начали решать новые задачи путем создания новых протоколов для стека TCP/IP, таких как протокол резервирования ресурсов (RSVP), защищенный протокол IP (IPSec), протокол коммутации меток (MPLS) и т. п. Однако ведущим специалистам было ясно, что только за счет добавления новых протоколов технологию TCP/IP развивать нельзя — нужно решиться на *модернизацию сердцевины стека*, протокола IP. Некоторые проблемы нельзя было решить без изменения формата IP-пакета и логики обработки полей заголовка IP-пакетов. Наиболее очевидной проблемой такого рода была проблема дефицита IP-адресов, которую невозможно снять, не расширив размер полей адресов источника и приемника.

Критике стала все чаще подвергаться схема масштабирования маршрутизации. Дело в том, что быстрый рост сети вызвал перегрузку маршрутизаторов, которым приходится обрабатывать в своих таблицах маршрутизации информацию о нескольких десятках тысяч номеров сетей, да еще решать некоторые вспомогательные задачи, такие, например, как фрагментация пакетов. Некоторые из предлагаемых решений данной проблемы также требовали внесения изменений в протокол IP.

Наряду с добавлением новых функций непосредственно в протокол IP необходимо было обеспечить его тесное взаимодействие с новыми протоколами — членами стека TCP/IP, что также требовало добавления в заголовок IP новых полей, обработку которых осуществляли бы эти протоколы. Например, для работы RSVP было желательно введение в заголовок IP поля метки потока, а для протокола IPSec — специальных полей для передачи данных, поддерживающих его функции обеспечения безопасности.

В результате сообщество Интернета после достаточно долгого обсуждения решило подвергнуть протокол IP серьезной переработке[1], выбрав в качестве основных целей модернизации:

- создание масштабируемой схемы адресации;
- сокращение объема работы, выполняемой маршрутизаторами;
- предоставление гарантий качества транспортных услуг;
- обеспечение защиты данных, передаваемых по сети.

Система адресации протокола IPv6

Новая (шестая) версия протокола IP (IPv6) внесла существенные изменения в систему адресации. Прежде всего это коснулось увеличения разрядности адреса: вместо 4 байт IP-адреса в версии IPv4 в новой версии под адрес отведено *16 байт*. Это дает возможность пронумеровать огромное количество узлов:

340 282 366 920 938 463 463 374 607 431 762 211 456.

Масштаб этого числа иллюстрирует, например, такой факт: если разделить это теоретически возможное количество IP-адресов между всеми жителями Земли (а их сегодня примерно 6 миллиардов), то на каждого из них придется невообразимо, если не сказать бессмысленно, большое количество IP-адресов — $5{,}7 \times 10^{28}$! Очевидно, что такое значи-

[1] В августе 1998 года были приняты пересмотренные версии группы стандартов, определяющих как общую архитектуру протокола IPv6 (RFC 2460), так и его отдельные аспекты, например систему адресации (RFC 4291).

тельное увеличение длины адреса было сделано не только и даже не столько для снятия проблемы дефицита адресов.

Главной целью изменения системы адресации было не механическое увеличение адресного пространства, а повышение эффективности работы стека TCP/IP в целом.

Вместо прежних двух уровней иерархии адреса (номер сети и номер узла) в IPv6 определено четыре уровня, из которых три служат для идентификации сетей, а один — для идентификации узлов сети. В новой версии не поддерживаются классы адресов (A, B, C, D, E), но широко используется технология CIDR. Благодаря этому, а также усовершенствованной системе групповой адресации и введению адресов нового типа IPv6 позволяет *сократить затраты на маршрутизацию*.

Произошли и чисто внешние изменения — разработчики стандарта предложили использовать вместо десятичной *шестнадцатеричную* форму записи IP-адреса. Каждые четыре шестнадцатеричные цифры отделяются друг от друга двоеточием. Вот как, например, может выглядеть адрес IPv6: FEDC:0A98:0:0:0:0:7654:3210. Для сетей, поддерживающих обе версии протокола (IPv4 и IPv6), разрешается задействовать для младших четырех байт традиционную для IPv4 десятичную запись: 0:0:0:0:0:FFFF:129.144.52.38.

В новой версии IPv6 предусмотрено три основных типа адресов:

- *Индивидуальный адрес* (unicast) является уникальным идентификатором отдельного интерфейса конечного узла или маршрутизатора. Назначение этого типа адреса совпадает с назначением уникальных адресов в версии IPv4. В то же время в версии IPv6 в отличие от версии IPv4 отсутствует понятие класса сети (A, B, C и D) и связанное с ним фиксированное разбиение адреса на номера сети и узла по границам байтов.
- *Групповой адрес* (multicast) аналогичен по назначению групповому адресу IPv4 — он идентифицирует группу интерфейсов, относящихся, как правило, к разным узлам. Пакет с таким адресом доставляется *всем* интерфейсам, имеющим такой адрес. В версии IPv6 групповой адрес имеет признак обзора (scope), отсутствовавший в групповом адресе версии IPv4. Этот признак позволяет гибко задавать область действия группового адреса, которая может представлять собой, например, только одну подсеть, только все подсети данного предприятия либо весь Интернет. Это упрощает работу маршрутизаторов, которым необходимо выявлять все узлы, относящиеся к какой-либо группе.
- *Адрес произвольной рассылки* (anycast) — это новый тип IP-адреса, определяющий группу интерфейсов. Но в отличие от группового адреса пакет, в поле адреса назначения которого стоит адрес произвольной рассылки, доставляется *одному* из интерфейсов группы, как правило, «ближайшему», в соответствии с метрикой, используемой протоколами маршрутизации. Синтаксически адрес произвольной рассылки ничем не отличается от индивидуального адреса, он назначается из того же диапазона адресов, что и индивидуальные адреса. Адрес произвольной рассылки может быть назначен только интерфейсам маршрутизатора. Интерфейсы маршрутизаторов, входящие в одну группу адресов произвольной рассылки, имеют индивидуальные адреса и, кроме того, общий адрес произвольной рассылки. Адреса такого типа ориентированы на *маршрутизацию от источника*, когда маршрут прохождения пакета определяется узлом-отправителем путем указания IP-адресов всех промежуточных маршрутизаторов. Например, провайдер может присвоить всем своим маршрутизаторам один и тот же адрес произвольной

рассылки и сообщить его абонентам. Если абонент желает, чтобы его пакеты передавались через сеть этого провайдера, то ему достаточно указать этот адрес произвольной рассылки в цепочке адресов маршрута от источника и пакет будет передан через ближайший маршрутизатор этого провайдера.

Тип адреса определяется значением нескольких старших битов адреса, которые названы **префиксом формата**.

Индивидуальные адреса делятся на несколько подтипов, основным среди которых является **глобальный агрегируемый уникальный адрес**. В отличие от уникальных адресов узлов версии IPv4, которые состоят из двух полей — номеров сети и узла, — глобальные агрегируемые уникальные адреса IPv6 имеют более сложную структуру, включающую шесть полей (рис. 15.25).

3	13	8	24	16	64
FP	TLA		NLA	SLA	Идентификатор интерфейса

Рис. 15.25. Структура глобального агрегируемого уникального адреса в пакете IPv6

- **Префикс формата** (Format Prefix, FP) для этого типа адресов имеет размер 3 бита и значение 001.
- **Поле TLA** (Top-Level Aggregation) предназначено для идентификации сетей самых крупных поставщиков услуг. Конкретное значение этого поля представляет собой общую часть адресов, которыми располагает данный поставщик услуг. Сравнительно небольшое количество разрядов, отведенных под это поле (13), выбрано специально для ограничения размера таблиц маршрутизации в магистральных маршрутизаторах самого верхнего уровня Интернета. Это поле позволяет перенумеровать 8196 сетей поставщиков услуг верхнего уровня, а значит, число записей, описывающих маршруты между этими сетями, также будет ограничено значением 8196, что ускорит работу магистральных маршрутизаторов. Следующие 8 разрядов зарезервированы на будущее для расширения при необходимости поля TLA.
- **Поле NLA** (Next-Level Aggregation) предназначено для нумерации сетей средних и мелких поставщиков услуг. Значительный размер поля NLA позволяет путем агрегирования адресов отразить многоуровневую иерархию поставщиков услуг.
- **Поле SLA** (Site-Level Aggregation) предназначено для адресации подсетей отдельного абонента, например подсетей одной корпоративной сети.
- **Идентификатор интерфейса** является аналогом номера узла в IPv4. Отличием версии IPv6 является то, что в общем случае идентификатор интерфейса просто *совпадает с его локальным* (*аппаратным*) *адресом*, а не представляет собой произвольно назначенный администратором номер узла. Идентификатор интерфейса имеет длину 64 бита, что позволяет поместить туда MAC-адрес (48 бит), адрес конечного узла ATM (48 бит) или номер виртуального соединения ATM (до 28 бит), а также, вероятно, даст возможность использовать локальные адреса технологий, которые могут появиться в будущем. Такой подход *делает ненужным протокол ARP*, поскольку процедура отображения IP-адреса на локальный адрес становится тривиальной — она сводится к простому отбрасыванию старшей части адреса. Кроме того, в большинстве случаев *отпадает необходимость*

ручного конфигурирования конечных узлов, так как младшую часть адреса — идентификатор интерфейса — узел узнает от аппаратуры (сетевого адаптера и т. п.), а старшую — номер подсети — ему сообщает маршрутизатор.

Рассмотрим пример (рис. 15.26). Пусть клиент получил от поставщика услуг пул адресов IPv6, определяемый префиксом 20:0A:00:C9:74:05/48. Поскольку первые 3 бита этого числа равны 001, это *глобальный агрегируемый уникальный адрес*.

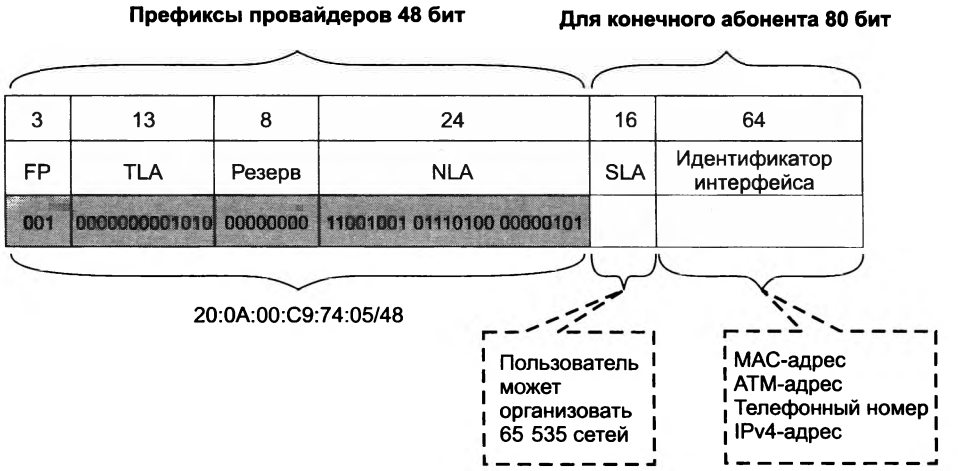


Рис. 15.26. Пример глобального агрегируемого адреса

Адрес этот принадлежит поставщику услуг верхнего уровня, у которого все сети имеют префикс 20:0A/16. Он может выделить поставщику услуг второго уровня некоторый диапазон адресов с общим префиксом, образованным его собственным префиксом, а также частью поля NLA. Длина поля NLA, отводимая под префикс, определяется маской, которую поставщик услуг верхнего уровня также должен сообщить своему клиенту — поставщику услуг второго уровня. Пусть в данном примере маска состоит из 32 единиц в старших разрядах, а результирующий префикс поставщика услуг второго уровня имеет вид 20:0A:00:C9/32.

В распоряжении поставщика услуг второго уровня остается 16 разрядов поля NLA для нумерации сетей своих клиентов. В качестве клиентов могут выступать поставщики услуг третьего и более низких уровней, а также конечные абоненты — предприятия и организации. Пусть, например, следующий байт (01110100) в поле NLA поставщик услуг использовал для передачи поставщику услуг более низкого (третьего) уровня, а тот, в свою очередь, использовал последний байт поля NLA для назначения пула адресов клиенту. Таким образом, с участием поставщиков услуг трех уровней был сформирован префикс 20:0A:00:C9:74:05/48, который получил клиент.

Протокол IPv6 оставляет в полном распоряжении клиента 2 байта (поле SLA) для нумерации сетей и 8 байт (поле идентификатора интерфейса) для нумерации узлов. Имея такой огромный диапазон номеров подсетей, администратор получает широкие возможности. Для сравнительно небольшой сети он может выбрать плоскую организацию, назначая каждой имеющейся подсети произвольные неповторяющиеся значения из диапазона

в 65 535 адресов, игнорируя оставшиеся. В крупных сетях более эффективным способом (сокращающим размеры таблиц корпоративных маршрутизаторов) может оказаться иерархическая структуризация сети на основе *агрегирования адресов*. В этом случае используется та же технология CIDR, но уже не поставщиком услуг, а администратором корпоративной сети.

ПРИМЕЧАНИЕ

Очевидно, что при таком изобилии сетей, которое предоставляется клиенту в IPv6, совершенно теряет смысл операция использования масок для разделения сетей на подсети, в то время как обратная процедура — объединение подсетей — приобретает особое значение. Разработчики стандартов IPv6 считают, что агрегирование адресов является основным способом эффективного расходования адресного пространства в новой версии протокола IP.

Снижение нагрузки на маршрутизаторы

Одной из основных целей изменения формата заголовка протокола IPv6 было сокращение накладных расходов, то есть уменьшение объема служебной информации, передаваемой с каждым пакетом. Для этого в новом протоколе IP были введены понятия основного и дополнительных заголовков. Основной заголовок присутствует всегда, а необязательные дополнительные заголовки могут содержать, например, информацию о фрагментации исходного пакета, полный маршрут следования пакета при маршрутизации от источника, информацию, необходимую для защиты передаваемых данных.

Основной заголовок имеет фиксированную длину в 40 байт, его формат показан на рис. 15.27.

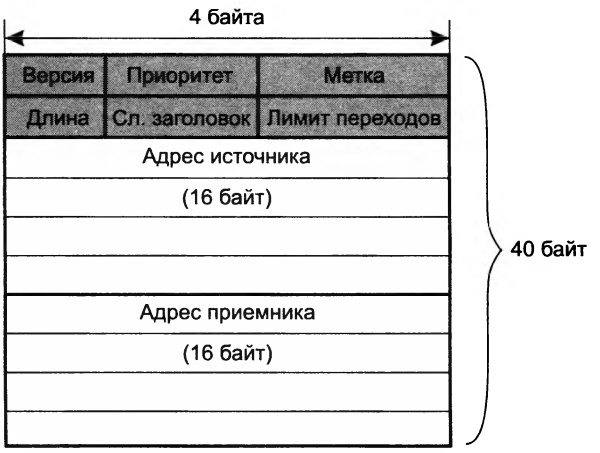


Рис. 15.27. Формат основного заголовка

Поле следующего заголовка соответствует по назначению полю протокола в версии IPv4 и содержит данные, определяющие тип заголовка, который следует за текущим. Каждый следующий дополнительный заголовок также содержит поле следующего заголовка. Если IP-пакет не содержит дополнительных заголовков, то в этом поле будет

значение, закрепленное за протоколом TCP, UDP, RIP, OSPF или другим, определенным в стандарте IPv4.

В предложениях по поводу протокола IPv6 фигурируют пока следующие типы дополнительных заголовков:

- **заголовок маршрутизации** — указание полного маршрута при маршрутизации от источника;
- **заголовок фрагментации** — информация, относящаяся к фрагментации IP-пакета (поле обрабатывается только в конечных узлах);
- **заголовок аутентификации** — информация, необходимая для аутентификации конечных узлов и обеспечения целостности содержимого IP-пакетов;
- **заголовок системы безопасности** — информация, необходимая для обеспечения конфиденциальности передаваемых данных путем шифрования и дешифрирования;
- **специальные параметры** — параметры, необходимые для последовательной обработки пакетов на каждом маршрутизаторе;
- **параметры получателя** — дополнительная информация для узла назначения.

Таким образом, IP-пакет может иметь, например, формат, показанный на рис. 15.28.

Рис. 15.28. Структура пакета IPv6

Поскольку для маршрутизации пакета обязательным является лишь основной заголовок (почти все дополнительные заголовки обрабатываются только в конечных узлах), *это снижает нагрузку на маршрутизаторы*. В то же время возможность использования большого количества дополнительных параметров *расширяет функциональность протокола IP* и делает его открытым для внедрения новых механизмов.

Для того чтобы повысить производительность маршрутизаторов Интернета в части выполнения их основной функции — продвижения пакетов, в версии IPv6 предпринят ряд мер по освобождению маршрутизаторов от некоторых вспомогательных функций.

- *Перенесение функций фрагментации с маршрутизаторов на конечные узлы.* Конечные узлы в версии IPv6 обязаны найти минимальное значение MTU вдоль всего пути, соединяющего исходный узел с узлом назначения (эта техника под названием Path MTU Discovery уже используется в IPv4). Маршрутизаторы IPv6 не выполняют фрагментацию, а только посылают ICMP-сообщение о слишком длинном пакете конечному узлу, который должен уменьшить размер пакета.
- *Агрегирование адресов* ведет к уменьшению размера адресных таблиц маршрутизаторов, а значит, к сокращению времени просмотра и обновления таблиц. При этом также сокращается служебный трафик, порождаемый протоколами маршрутизации.
- *Широкое использование маршрутизации от источника.* При маршрутизации от источника узел-источник задает полный маршрут прохождения пакета через сети. Такая техника освобождает маршрутизаторы от необходимости просмотра адресных таблиц при выборе следующего маршрутизатора.
- *Отказ от обработки необязательных параметров заголовка.*
- *Использование в качестве номера узла его MAC-адреса* избавляет маршрутизаторы от необходимости применять протокол ARP.

Новая версия протокола IP, являющаяся составной частью проекта IPv6, предлагает встроенные средства защиты данных. Размещение средств защиты на сетевом уровне делает их прозрачными для приложений, так как между уровнем IP и приложением всегда будет работать протокол транспортного уровня. Приложения переписывать при этом не придется. Новая версия протокола IP со встроенными средствами обеспечения безопасности называется **IPSec** (Security Internet Protocol — защищенный протокол IP). Возможности этого протокола подробно рассматриваются в главе 29.

Переход на версию IPv6

При разработке IPv6 была предусмотрена возможность плавного перехода к новой версии, когда довольно значительное время будут сосуществовать островки Интернета, работающие по протоколу IPv6, и остальная часть Интернета, работающая по протоколу IPv4. Существует несколько подходов к организации взаимодействия узлов, использующих разные стеки TCP/IP.

Трансляция протоколов. Трансляция протоколов реализуется шлюзами, которые устанавливаются на границах сетей, использующих разные версии протокола IP. Согласование двух версий протокола IP происходит путем преобразования пакетов IPv4 в IPv6 и наоборот. Процесс преобразования включает, в частности, отображение адресов сетей и узлов, различным образом трактуемых в этих протоколах. Для упрощения преобразования адресов между версиями разработчики IPv6 предлагают использовать специальные подтип IPv6-адреса — **IPv4-совместимый IPv6-адрес**, который в младших четырех байтах переносит IPv4-адрес, а в старших 12 байтах содержит нули (рис. 15.29). Это позволяет получать IPv4-адрес из IPv6-адреса простым отбрасыванием старших байтов.

Для решения обратной задачи — передачи пакетов IPv4 через части Интернета, работающие по протоколу IPv6, — предназначен **IPv4-отображенный IPv6-адрес**. Этот тип адреса также содержит в 4 младших байтах IPv4-адрес, в старших 10-ти байтах — нули, а в 5-м и 6-м байтах IPv6-адреса — единицы, которые показывают, что узел поддерживает только версию 4 протокола IP (рис. 15.30).

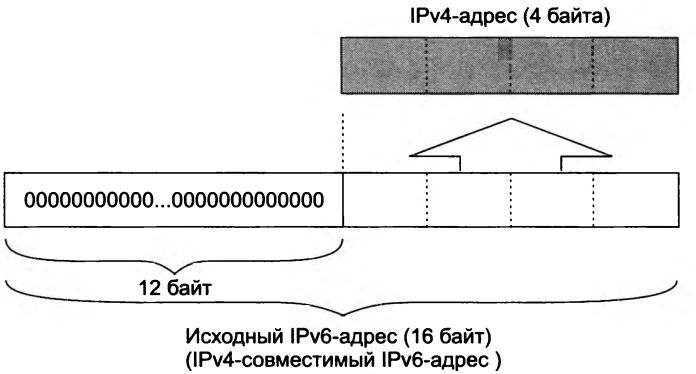


Рис. 15.29. Преобразование IPv6 в IPv4

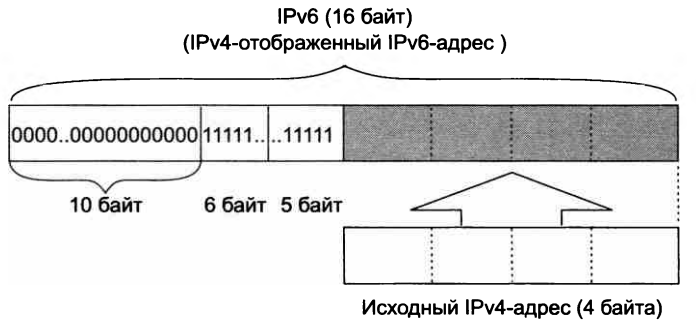


Рис. 15.30. Преобразование IPv4 в IPv6

- *Мультиплексирование стеков протоколов* означает установку на взаимодействующих хостах сети обеих версий протокола IP. Обе версии стека протоколов должны быть развернуты также на разделяющих эти хосты маршрутизаторах. В том случае, когда IPv6-хост отправляет сообщение IPv6-хосту, он использует стек IPv6, а если тот же хост взаимодействует с IPv4-хостом — стек IPv4. Маршрутизатор с установленными на нем двумя стеками называется маршрутизатором IPv4/IPv6, он способен обрабатывать трафики разных версий независимо друг от друга. В настоящее время практически все операционные системы оснащены полнофункциональными версиями стеков IPv4 и IPv6 и мультиплексирование стеков стало основным подходом к организации их совместного использования.
- *Инкапсуляция*, или *туннелирование*. Инкапсуляция — это еще один метод решения задачи согласования сетей, использующих разные версии протокола IP. Инкапсуляция может быть применена, когда две сети одной версии протокола, например IPv4, необходимо соединить через транзитную сеть, работающую по другой версии, например IPv6 (рис. 15.31). При этом пакеты IPv4 помещаются в пограничных устройствах (на рисунке роль согласующих устройств исполняют маршрутизаторы) в пакеты IPv6 и переносятся через «туннель», проложенный в IPv6-сети. Такой способ имеет недостаток, заключающийся в том, что узлы IPv4-сетей не имеют возможности взаимодействовать с узлами транзитной IPv6-сети. Аналогичным образом метод туннелирования может использоваться для переноса пакетов IPv6 через сеть маршрутизаторов IPv4.

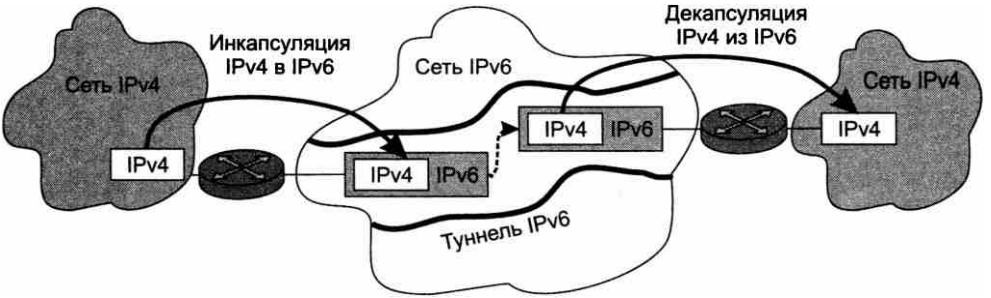


Рис. 15.31. Согласование технологий IPv4 и IPv6 путем туннелирования (инкапсуляции)

Переход от версии IPv4 к версии IPv6 уже идет «полным ходом». Например, по данным на 21 марта 2014 года, доступ по протоколу IPv6 к веб-сайтам Google в среднем получают 5 % пользователей. И хотя эти 5 % распределены по странам очень неравномерно — 32 % Бельгия, 16 % США, 14 % Германия, ..., 0,56 % Россия, 0,34 % Великобритания, — качественная картина успешного продвижения версии IPv6 выглядит вполне убедительно.

Выводы

Протокол IP решает задачу доставки сообщений между узлами составной сети. Поскольку он является дейтаграммным, никаких гарантий надежной доставки сообщений не дается.

Максимальная длина IP-пакета составляет 65 535 байт. Заголовок обычно имеет длину 20 байт и содержит информацию о сетевых адресах отправителя и получателя, параметры фрагментации, время жизни пакета, контрольную сумму и некоторые другие параметры.

Вид таблицы IP-маршрутизации зависит от конкретной реализации маршрутизатора. Несмотря на значительные внешние различия выводимых на экран таблиц, все они включают два обязательных поля — это поля адреса назначения и следующего маршрутизатора.

Записи в таблицу маршрутизации могут поступать из разных источников. Во-первых, в результате конфигурирования программное обеспечение стека TCP/IP заносит в таблицу записи о непосредственно подключенных сетях и маршрутизаторах по умолчанию, а также записи об особых адресах. Во-вторых, администратор вручную заносит записи о специфических маршрутах и о маршруте по умолчанию. В-третьих, протоколы маршрутизации автоматически заносят в таблицу динамические записи об имеющихся маршрутах.

Эффективным средством структуризации IP-сетей являются маски. Маски позволяют разделить одну сеть на несколько подсетей или объединить несколько сетей в одну более крупную сеть.

Значительная роль в будущих IP-сетях отводится технологии бесклассовой междоменной маршрутизации (CIDR), которая решает две основные задачи. Первая состоит в более экономном расходовании адресного пространства, вторая — в уменьшении числа записей в таблицах.

Важной особенностью протокола IP, отличающей его от других сетевых протоколов, например от сетевого протокола IPX, является его способность выполнять динамическую фрагментацию пакетов при передаче их между сетями с различными максимально допустимыми значениями длины поля данных кадров (MTU).

Протокол ICMP играет в сети вспомогательную роль. Он используется для диагностики и мониторинга сети. Так, в основе работы популярных утилит мониторинга IP-сетей ping и tracert лежат ICMP-сообщения.

Новая версия протокола IP, являющаяся составной частью проекта IPv6, направлена на создание масштабируемой схемы адресации, сокращение объема работы, выполняемой маршрутизаторами, предоставление гарантий качества транспортных услуг, а также обеспечение защиты данных, передаваемых по сети.

Контрольные вопросы

1. Может ли один сетевой интерфейс иметь одновременно несколько IPv6-адресов разных типов: уникальный адрес, адрес произвольной рассылки, групповой адрес?
2. Передается ли в IP-пакете маска в тех случаях, когда маршрутизация реализуется с использованием масок?
3. Какие элементы сети могут выполнять фрагментацию? Варианты ответов:
   а) только компьютеры;
   б) только маршрутизаторы;
   в) компьютеры, маршрутизаторы, мосты, коммутаторы;
   г) компьютеры и маршрутизаторы.
4. Что произойдет, если при передаче пакета он был фрагментирован и один из фрагментов не дошел до узла назначения после истечения тайм-аута? Варианты ответов:
   а) IP-модуль получателя сообщит о неполучении одного фрагмента, а IP-модуль узла-отправителя повторит передачу недошедшего фрагмента;
   б) IP-модуль получателя сообщит о неполучении одного фрагмента, а IP-модуль узла-отправителя повторит передачу всего пакета, в состав которого входил недошедший фрагмент;
   в) IP-модуль узла-получателя отбросит все полученные фрагменты пакета, в котором потерялся один фрагмент, а IP-модуль узла-отправителя не будет предпринимать никаких действий по повторной передаче данного пакета.
5. Кому адресовано ICMP-сообщение? Варианты ответов:
   а) протоколу IP узла-отправителя пакета, вызвавшего ошибку;
   б) протоколу IP ближайшего маршрутизатора, от которого поступил пакет, вызвавший ошибку;
   в) протоколу транспортного или прикладного уровня узла-отправителя пакета, вызвавшего ошибку.
6. В разделе «Перекрытие адресных пространств» этой главы приведен пример того, как администратор планирует сеть своего предприятия. Решите ту же задачу по планированию сети для случая, когда для сети Ethernet требуется 300 адресов, для сети Token Ring — 30, для DMZ —20 и для соединительной сети — 8. Какой пул адресов необходимо получить у поставщика услуг на этот раз? (Для определенности будем считать, что поставщик услуг выделит непрерывный пул адресов.) Как администратор распределит адреса между своими четырьмя сетями? Как будут выглядеть таблицы маршрутизации R1 и R2?

ГЛАВА 16 Протоколы транспортного уровня TCP и UDP

Мультиплекирование и демультиплексирование приложений

Порты

Каждый компьютер может выполнять несколько процессов, более того, даже отдельный прикладной процесс может иметь несколько точек входа, выступающих в качестве адресов назначения для пакетов данных. Поэтому доставка данных на сетевой интерфейс компьютера-получателя — это еще не конец пути, так как данные необходимо переправить конкретному процессу-получателю. Реализуемая протоколами TCP и UDP процедура распределения между прикладными процессами пакетов, поступающих от сетевого уровня, называется **демультиплексированием** (рис. 16.1).

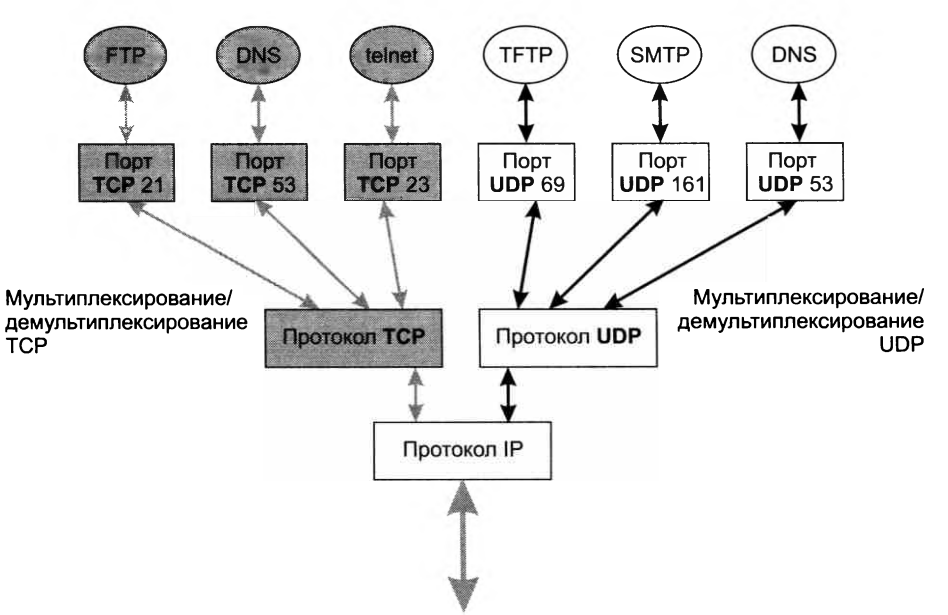


Рис. 16.1. Мультиплексирование и демультиплексирование на транспортном уровне

Существует и обратная задача: данные, генерируемые разными приложениями, работающими на одном конечном узле, должны быть переданы общему для всех них протокольному модулю IP для последующей отправки в сеть. Эту работу, называемую **мультиплексированием**, тоже выполняют протоколы TCP и UDP.

Протоколы TCP и UDP ведут для каждого приложения две системные очереди: очередь данных, поступающих к приложению из сети, и очередь данных, отправляемых этим приложением в сеть. Такие системные очереди называются **портами**[1], причем входная и выходная очереди одного приложения рассматриваются как один порт. Для идентификации портов им присваивают номера.

Если процессы представляют собой популярные системные службы, такие как FTP, telnet, HTTP, TFTP, DNS и т. п., то за ними закрепляются **стандартные назначенные номера**, называемые также **хорошо известными** (well-known) номерами портов. Эти номера закрепляются и публикуются в стандартах Интернета (RFC 1700, RFC 3232). Так, номер 21 закреплен за серверной частью службы удаленного доступа к файлам FTP, а 23 — за серверной частью службы удаленного управления telnet. Назначенные номера из диапазона от 0 до 1023 являются *уникальными в пределах Интернета* и закрепляются за приложениями *централизованно*.

Для тех приложений, которые еще не стали столь распространенными, номера портов назначаются *локально* разработчиками этих приложений или операционной системой в ответ на поступление запроса от приложения. На каждом компьютере операционная система ведет список занятых и свободных номеров портов. При поступлении запроса от приложения, выполняемого на данном компьютере, операционная система выделяет ему первый свободный номер. Такие номера называют **динамическими**. В дальнейшем все сетевые приложения должны адресоваться к данному приложению с указанием назначенного ему динамического номера порта. После того как приложение завершит работу, его номер возвращается в список свободных и может быть назначен другому приложению. Динамические номера являются *уникальными в пределах каждого компьютера*, но при этом обычной является ситуация совпадения номеров портов приложений, выполняемых на разных компьютерах. Как правило, клиентские части известных приложений (DNS, WWW, FTP, telnet и др.) получают динамические номера портов от ОС.

Все, что было сказано о портах, в равной степени относится к обоим протоколам транспортного уровня (TCP и UDP). В принципе, нет никакой зависимости между назначением номеров портов для приложений, использующих протокол TCP, и приложений, работающих с протоколом UDP. Приложения, которые передают данные на уровень IP по протоколу UDP, получают номера, называемые **UDP-портами**. Аналогично, приложениям, обращающимся к протоколу TCP, выделяются **TCP-порты**.

В том и в другом случаях это могут быть как назначенные, так и динамические номера. Диапазоны чисел, из которых выделяются номера TCP- и UDP-портов, совпадают: от 0 до 1023 для назначенных и от 1024 до 65 535 для динамических. Однако никакой связи между назначенными номерами TCP- и UDP-портов нет. Даже если номера TCP- и UDP-портов совпадают, они идентифицируют разные приложения. Например, одному приложению может быть назначен TCP-порт 1750, а другому — UDP-порт 1750. В некоторых случаях, когда приложение может обращаться по выбору к протоколу TCP или UDP (например, таким при-

[1] Порты приложений не надо путать с портами (сетевыми интерфейсами) оборудования.

ложением является DNS), ему, исходя из удобства запоминания, назначаются совпадающие номера TCP- и UDP-портов (в данном примере — это *хорошо известный* номер 53).

Сокеты

Стандартные назначенные номера портов уникально идентифицируют тип приложения (FTP, или HTTP, или DNS и т. д.), однако они не могут использоваться для однозначной идентификации прикладных процессов, связанных с каждым из этих типов приложений. Пусть, например, на одном хосте запущены две *копии* DNS-сервера — DNS-сервер 1, DNS-сервер 2 (рис. 16.2). Каждый из этих DNS-серверов имеет хорошо известный UDP-порт 53. Какому из этих серверов нужно было бы направить запрос клиента, если бы в DNS-запросе в качестве идентификатора сервера был указан только номер порта?

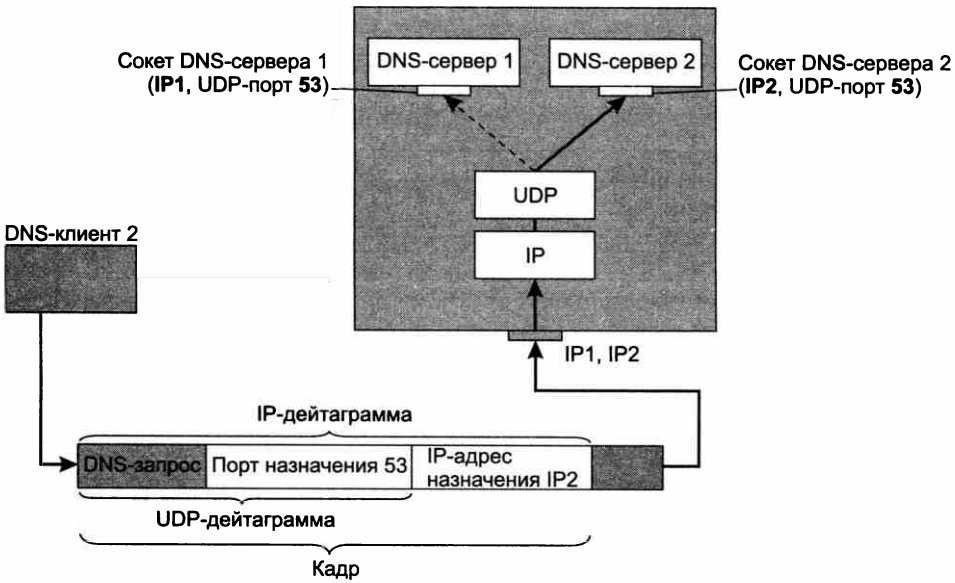


Рис. 16.2. Демультиплексирование протокола UDP на основе сокетов

Чтобы снять неоднозначность в идентификации приложений, разные копии связываются с разными IP-адресами. Для этого сетевой интерфейс компьютера, на котором выполняется несколько копий приложения, должен иметь соответствующее число IP-адресов — на рисунке это IP1 и IP2. Во всех IP-пакетах, направляемых DNS-серверу 1, в качестве IP-адреса указывается IP1, а в пакетах, направляемых на DNS-серверу 2, — адрес IP2. Поэтому показанный на рисунке пакет, в поле данных которого содержится UDP-дейтаграмма с указанным номером порта 53, а в поле заголовка задан адрес IP2, однозначно будет направлен заданному адресату — DNS-серверу 2.

Прикладной процесс однозначно определяется в пределах сети и в пределах отдельного компьютера парой (IP-адрес, номер порта), называемой **сокетом** (socket). Сокет, определенный IP-адресом и номером UDP-порта, называется **UDP-сокетом**, а IP-адресом и номером TCP-порта — **TCP-сокетом**.

Здесь мы должны уточнить описанную в предыдущих главах упрощенную картину прохождения пакета вверх по стеку. Действительно, как мы и отмечали, после получения IP-пакета от протокола канального уровня протокол IP анализирует содержимое заголовка этого пакета, после чего заголовок отбрасывается, и «наверх» передается содержимое поля данных IP-пакета, например UDP-дейтаграмма. Упрощение состоит в том, что вместе с содержимым поля данных на транспортный уровень передается извлеченный из заголовка IP-адрес назначения, который и используется для однозначной идентификации приложения.

Протокол UDP и UDP-дейтаграммы

Протокол UDP, подобно IP, является дейтаграммным протоколом, реализующим так называемый ненадежный сервис *по возможности*, который не гарантирует доставку сообщений адресату.

При работе на хосте-отправителе данные от приложений поступают протоколу UDP через порт в виде сообщений (рис. 16.3). Протокол UDP добавляет к каждому отдельному сообщению свой 8-байтный заголовок, формируя из этих сообщений собственные протокольные единицы, называемые **UDP-дейтаграммами**, и передает их нижележащему протоколу IP. В этом и заключаются его функции по *мультиплексированию* данных.

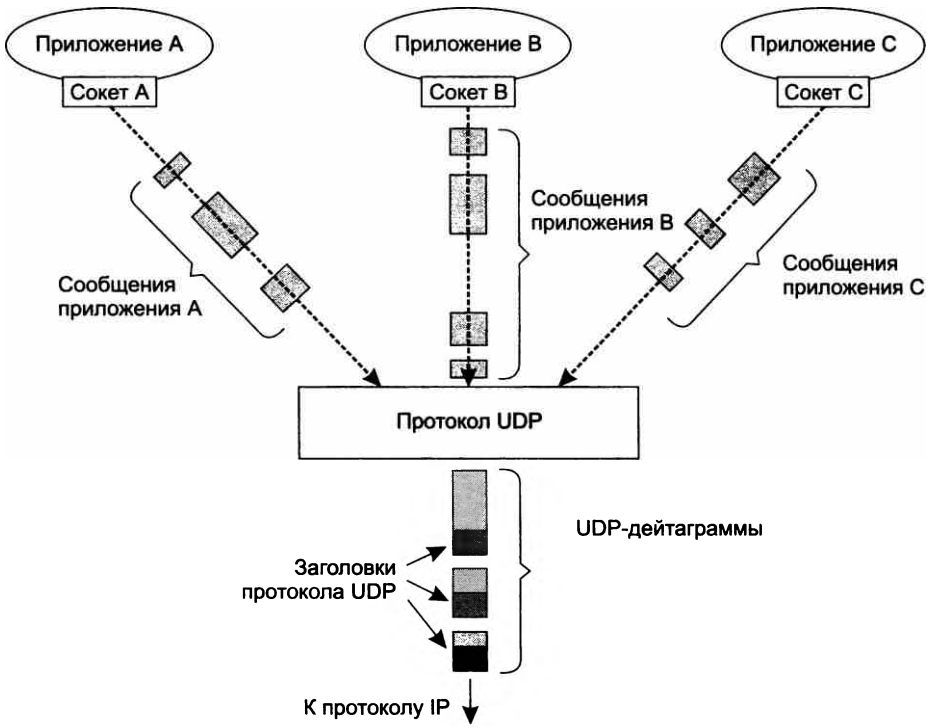


Рис. 16.3. Работа протокола UDP на хосте-отправителе

Каждая дейтаграмма переносит *отдельное пользовательское сообщение*. Сообщения могут иметь разную длину, не превышающую, однако, длину поля данных протокола IP, которое, в свою очередь, ограничено размером кадра технологии нижнего уровня. Поэтому если буфер UDP переполняется, то сообщение приложения отбрасывается.

Заголовок UDP состоит из четырех 2-байтных полей:

- номер UDP-порта отправителя;
- номер UDP-порта получателя;
- контрольная сумма;
- длина дейтаграммы.

Далее приведен пример заголовка UDP с заполненными полями:

```
Source Port = 0x0035
Destination Port = 0x0411
Total length = 132 (0x84) bytes
Checksum = 0x5333
```

В этой UDP-дейтаграмме в поле данных, длина которого, как следует из заголовка, равна (132 – 8) байт, помещено сообщение DNS-сервера, что можно видеть по номеру порта источника (`Source Port = 0x0035`). В шестнадцатеричном формате это значение равно стандартному номеру порта DNS-сервера — 53.

Судя по простоте заголовка, протокол UDP несложен. Действительно, его функции сводятся к простой передаче данных между прикладным и сетевым уровнями, а также к примитивному контролю искажений в передаваемых данных. При контроле искажений протокол UDP только *диагностирует, но не исправляет ошибку*. Если контрольная сумма показывает, что в поле данных UDP-дейтаграммы произошла ошибка, протокол UDP просто отбрасывает поврежденную дейтаграмму.

Работая на хосте-получателе, протокол UDP принимает от протокола IP извлеченные из пакетов UDP-дейтаграммы. Полученные из IP-заголовка IP-адрес назначения и из UDP-заголовка номер порта используются для формирования UDP-сокета, однозначно идентифицирующего приложение, которому направлены данные. Протокол UDP освобождает дейтаграмму от UDP-заголовка. Полученное в результате сообщение он передает приложению на соответствующий UDP-сокет. Таким образом, протокол UDP выполняет *демультиплексирование* на основе сокетов.

Протокол TCP и TCP-сегменты

Протокол TCP предназначен для передачи данных между приложениями. Этот протокол основан на *логическом соединении*, что позволяет ему обеспечивать гарантированную доставку данных, используя в качестве инструмента ненадежный дейтаграммный сервис протокола IP.

При работе на хосте-отправителе протокол TCP рассматривает информацию, поступающую к нему от прикладных процессов, как *неструктурированный поток байтов* (рис. 16.4). Поступающие данные буферизуются средствами TCP. Для передачи на

сетевой уровень из буфера «вырезается» некоторая непрерывная часть данных, которая называется **сегментом**[1] и снабжается заголовком.

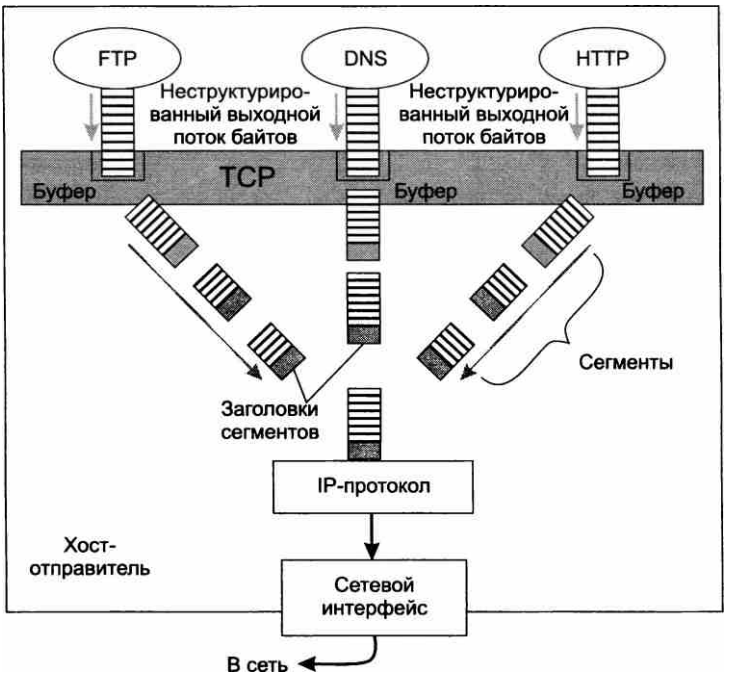


Рис. 16.4. Формирование TCP-сегментов из потока байтов

ПРИМЕЧАНИЕ

В отличие от протокола UDP, который создает свои дейтаграммы на основе логически обособленных единиц данных — сообщений, генерируемых приложениями, протокол TCP делит поток данных на сегменты без учета их смысла или внутренней структуры.

Заголовок TCP-сегмента содержит значительно больше полей, чем заголовок UDP, что отражает более развитые возможности протокола TCP (рис. 16.5). Краткие описания большинства полей помещены на рисунке, а более подробно мы их рассмотрим, когда будем изучать функции протокола TCP.

Коротко поясним значение однобитных полей, называемых **флагами**, или **кодовыми битами** (code bits). Они расположены сразу за резервным полем и содержат служебную информацию о типе данного сегмента. Положительное значение сигнализируется установкой этих битов в единицу:

- **URG** — срочное сообщение;
- **ACK** — квитанция на принятый сегмент;

[1] Заметим, что сегментом называют как единицу передаваемых данных в целом (поле данных и заголовок протокола TCP), так и отдельно поле данных.

- **PSH** — запрос на отправку сообщения без ожидания заполнения буфера;
- **RST** — запрос на восстановление соединения;
- **SYN** — сообщение, используемое для синхронизации счетчиков переданных данных при установлении соединения;
- **FIN** — признак достижения передающей стороной последнего байта в потоке передаваемых данных.

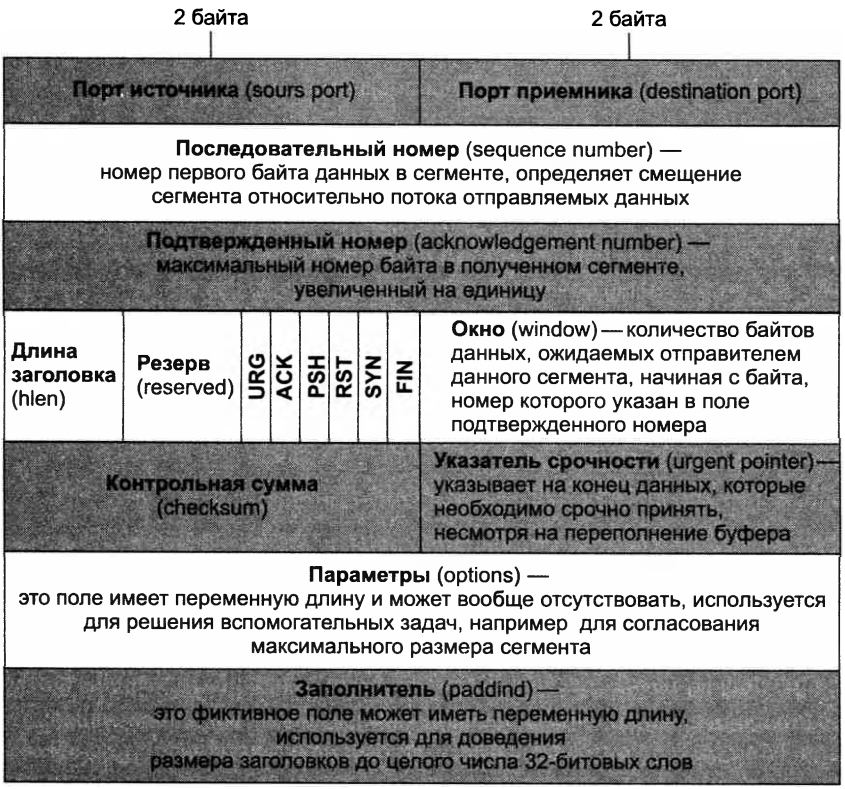


Рис. 16.5. Формат заголовка TCP-сегмента

Логические соединения — основа надежности TCP

Основным отличием TCP от UDP является то, что на протокол TCP возложена дополнительная задача — обеспечить *надежную доставку сообщений*, используя в качестве основы *ненадежный дейтаграммный протокол IP*.

Для решения этой задачи протокол TCP использует метод продвижения данных с установлением *логического соединения*. Как было сказано ранее, логическое соединение дает воз-

можность участникам обмена следить за тем, чтобы данные не были потеряны, искажены или продублированы, а также чтобы они пришли к получателю в том порядке, в котором были отправлены.

Далее в разделе «Методы квитирования» мы рассмотрим различные подходы к организации надежного логического канала, в том числе подробно остановимся на концепции *скользящего окна*, лежащей в основе протокола TCP.

Протокол TCP устанавливает логические соединения между *прикладными процессами*, причем в каждом соединении участвуют только *два* процесса. TCP-соединение является *дуплексным*, то есть каждый из участников этого соединения может одновременно получать и отправлять данные.

На рис. 16.6 показаны сети, соединенные маршрутизаторами, на которых установлен протокол IP. Установленные на конечных узлах протокольные модули TCP решают задачу обеспечения надежного обмена данными путем установления между собой **логических соединений**.

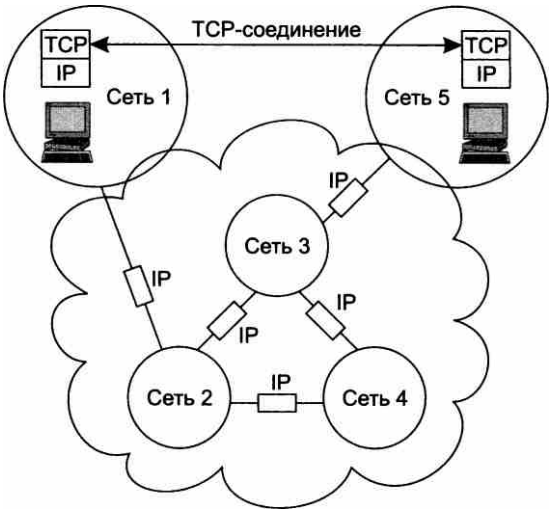


Рис. 16.6. TCP-соединение создает надежный логический канал между конечными узлами

При установлении логического соединения модули TCP договариваются между собой о параметрах процедуры обмена данными. В протоколе TCP каждая сторона соединения посылает противоположной стороне следующие параметры:

- максимальный размер сегмента, который она готова принимать;
- максимальный объем данных (возможно, несколько сегментов), которые она разрешает другой стороне передавать в свою сторону, даже если та еще не получила квитанцию на предыдущую порцию данных (размер окна);
- начальный порядковый номер байта, с которого она начинает отсчет потока данных в рамках данного соединения.

В результате переговорного процесса модулей TCP с двух сторон соединения определяются параметры соединения. Одни из них остаются постоянными в течение всего сеанса связи, а другие адаптивно изменяются.

Соединение устанавливается по инициативе клиентской части приложения. При необходимости выполнить обмен данными с серверной частью приложение-клиент обращается к нижележащему протоколу TCP, который в ответ на это обращение посылает сегмент-запрос на установление соединения протоколу TCP, работающему на стороне сервера (рис. 16.7, *а*). В числе прочего в запросе содержится флаг SYN, установленный в 1.

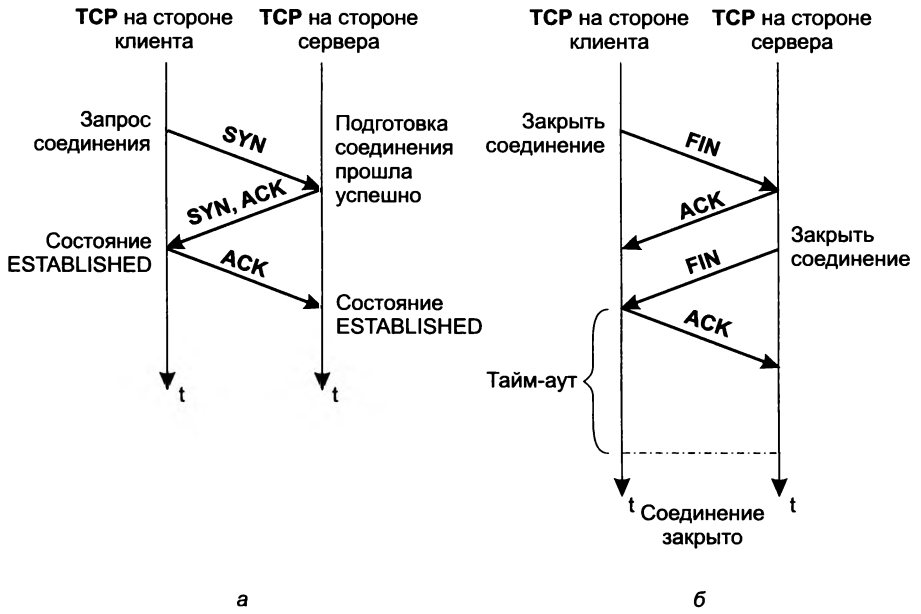


Рис. 16.7. Процедура установления и разрыва логического соединения при нормальном течении процесса

Получив запрос, модуль TCP на стороне сервера пытается создать «инфраструктуру» для обслуживания нового клиента. Он обращается к операционной системе с просьбой о выделении определенных системных ресурсов для организации буферов, таймеров, счетчиков. Эти ресурсы закрепляются за соединением с момента создания и до момента разрыва. Если на стороне сервера все необходимые ресурсы были получены и все необходимые действия выполнены, то модуль TCP посылает клиенту сегмент с флагами ACK и SYN.

В ответ клиент посылает сегмент с флагом ACK и переходит в состояние установленного логического соединения (состояние ESTABLISHED). Когда сервер получает флаг ACK, он также переходит в состояние ESTABLISHED. На этом процедура установления соединения заканчивается, и стороны могут переходить к обмену данными.

Соединение может быть разорвано в любой момент по инициативе любой стороны. Для этого клиент и сервер должны обменяться сегментами FIN и ACK, в последовательности, показанной на рис. 16.7, *б* (здесь инициатором является клиент). Соединение считается закрытым по прошествии некоторого времени, в течение которого сторона-инициатор убеждается, что ее завершающий сигнал ACK дошел нормально и не вызвал никаких «аварийных» сообщений со стороны сервера.

ПРИМЕЧАНИЕ

Мы описали здесь процедуры установления и закрытия соединения очень схематично. Реальные протокольные модули работают в соответствии с более сложными алгоритмами, учитывающими всевозможные «нештатные» ситуации, такие, например, как задержки и потери сегментов, недостаточность ресурсов или неготовность сервера к установлению соединения. Кроме того, мы проигнорировали тот факт, что еще на этапе установления соединения стороны договариваются о некоторых параметрах своего взаимодействия, например о начальных номерах посылаемых ими байтов. Однако мы скоро вернемся к этим важным деталям работы протокола TCP.

Логическое TCP-соединение однозначно идентифицируется парой сокетов, определенных для этого соединения двумя взаимодействующими процессами.

Сокет одновременно может участвовать в нескольких соединениях. Так, на рис. 16.8 показаны три компьютера с адресами IP1, IP2, IP3. На каждом компьютере выполняется по одному приложению — APPL1, APPL2 и APPL3, сокеты которых — соответственно (IP1, n1), (IP2, n2), (IP3, n3), а номера TCP-портов приложений — n1, n2, n3.

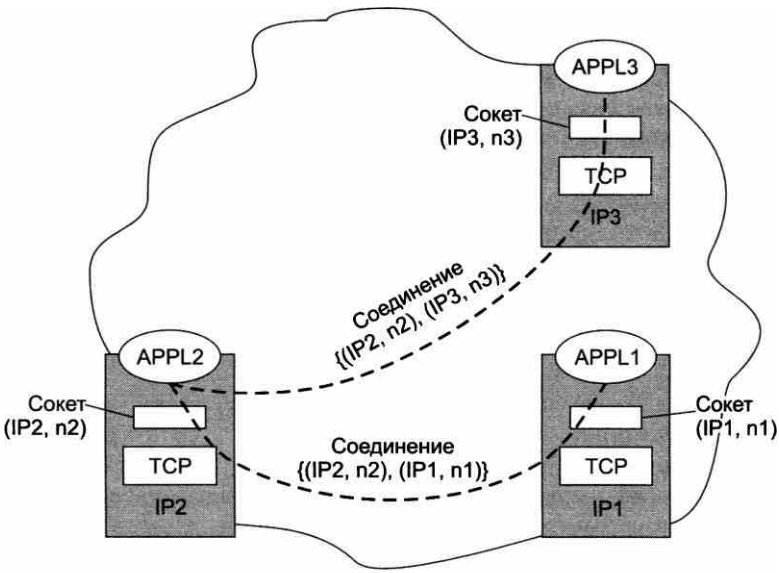


Рис. 16.8. Один сокет может участвовать в нескольких соединениях

На рисунке показаны два логических соединения, которые установило приложение 2 с приложением 1 и приложением 3. Логические соединения идентифицируются как {(IP2, n2), (IP1, n1)} и {(IP2, n2), (IP3, n3)} соответственно. Мы видим, что в обоих соединениях участвует один и тот же сокет — (IP2, n2).

А теперь рассмотрим на примере, как протокол TCP выполняет демультиплексирование. Пусть некий поставщик услуг оказывает услугу по веб-хостингу, то есть на его компьютере клиенты могут разворачивать свои веб-серверы. Работа веб-сервера основана на протоколе

прикладного уровня HTTP, который передает свои сообщения в TCP-сегментах. Модуль TCP ожидает запросы от веб-клиентов (браузеров), «прослушивая» хорошо известный порт 80.

На рис. 16.9 показан вариант хостинга с двумя веб-серверами — сервером www1.model.ru, имеющим IP-адрес IP1, и сервером www2.tour.ru с адресом IP2. К каждому из них может обращаться множество клиентов, причем клиенты могут одновременно работать как с сервером www1, так и с сервером www2. Для каждой пары клиент-сервер протоколом TCP создается *отдельное логическое соединение*.

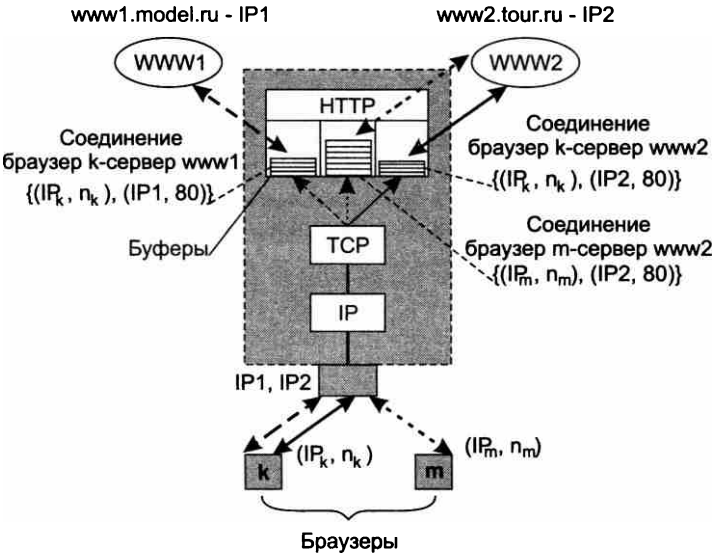


Рис. 16.9. Демультиплексирование протокола TCP на основе соединений

На рисунке показаны два браузера, имеющие соответственно сокеты (IP_k, n_k) и (IP_m, n_m). Пользователь браузера k обращается одновременно к серверам WWW1 и WWW2. Наличие отдельных соединений для работы с каждым из этих серверов обеспечивает не только надежную доставку, но и разделение информационных потоков — у пользователя никогда не возникает вопроса, каким сервером ему была послана та или иная страница. Одновременно с пользователем браузера k с сервером WWW2 работает пользователь браузера m. И в этом случае отдельные логические соединения, в рамках которых идет работа обоих пользователей, позволяют изолировать их информационные потоки. На рисунке показаны буферы, количество которых определяется не числом веб-серверов и не числом клиентов, а числом логических соединений. Сообщения в эти буферы направляются в зависимости от значений сокетов как отправителя, так и получателя. Отсюда можно сделать вполне конкретный вывод.

> Протокол TCP осуществляет демультиплексирование информации, поступающей на прикладной уровень, на основе *соединений* процессов или, что одно и то же, на основе идентифицирующих эти процессы *пар сокетов*.

Методы квитирования

Прежде чем перейти к изучению конкретных механизмов протокола TCP, которые он использует для обеспечения надежной передачи, мы попробуем шире взглянуть на эту проблему. Один из наиболее естественных приемов, используемых для организации надежного обмена данными, — это передача с квитированием, суть которой состоит в следующем.

> Отправитель отсылает данные, а получатель подтверждает их получение квитанциями. Если отправитель вовремя не получает квитанции на переданные данные, то он передает их *повторно*.

При всей простоте сформулированной схемы, за которой закрепилось особое название **запрос повторной передачи** (Automatic Repeat reQuest, **ARQ**), любая попытка ее реализации «обрастает» множеством деталей и вопросов. Например, должен ли отправитель ждать, пока не придет квитанция на отправленный пакет[1], прежде чем отсылать следующий? Должна ли принимающая сторона подтверждать приход каждого или сразу нескольких пакетов? Какое время ожидания квитанции источником следует считать предельно допустимым? Что, если квитанция потеряется и отправитель еще раз пошлет тот же пакет? Каким образом приемник должен распознавать дубликаты пакетов, а источник — квитанций? Различные протоколы передачи данных с квитированием по-разному отвечают на эти и другие вопросы, в результате данные протоколы отличаются производительностью, надежностью и объемами потребляемых ресурсов. Все многообразие возможных решений может быть разделено на два класса:

- методы **простоя источника** (Stop-and-Wait);
- методы **скользящего окна**.

В свою очередь, методы скользящего окна тоже подразделяются на два класса:

- методы, использующие *окно передачи*, — к ним, в частности, относится метод **передачи с возвращением на N пакетов** (Go-Back-N);
- методы, использующие *окно передачи и окно приема*, — примером является метод **передачи с выборочным повторением** (Selective Repeat).

Перечислим некоторые общие черты, присущие всем этим методам.

- Отправитель (источник) и получатель (приемник), в общем случае работающие *асинхронно*, осуществляют передачу пакетов по *ненадежной* линии связи, в которой возможны искажения, большие задержки и потери пакетов.
- Отправитель принимает данные от *протокола верхнего уровня* (приложения), получатель передает полученные данные на верхний уровень (приложению).
- Получатель располагает механизмом определения искаженных пакетов, например по контрольной сумме.
- После успешного получения пакета получатель посылает отправителю квитанции (acknowledgment, **ACK**).
- Для отслеживания задержек пакетов используется **таймер**, тайм-аут которого устанавливается равным предельному времени ожидания квитанции.

---

[1] В данном случае не имеет значения, какое название используется для единицы передаваемых данных — кадр, пакет или сообщение.

Метод простоя источника

В методе **простоя источника** отправитель передает последовательность ненумерованных пакетов. Метод требует, чтобы отправитель дожидался от получателя квитанции и только *после этого* посылал следующий пакет.

С переданным пакетом отправитель связывает таймер. Если в течение тайм-аута квитанция не пришла, то пакет (или квитанция на него) считается утерянным или искаженным и его передача повторяется. На рис. 16.10 показано, что второй пакет (здесь пакеты нумеруются только для нашего удобства, отправитель и получатель не имеют дела с номерами) отсылается только после того, как пришла квитанция, подтверждающая доставку первого пакета. Однако затем произошла длительная пауза в отправке очередного третьего пакета. В течение этой паузы источник, не дождавшись квитанции, которая была *утеряна*, послал повторно пакет 2, в результате получатель теперь имеет оригинальный пакет 2 и его дубликат. Понятно, что при таком алгоритме работы принимающая сторона должна уметь распознавать дублированные пакеты и отбрасывать их.

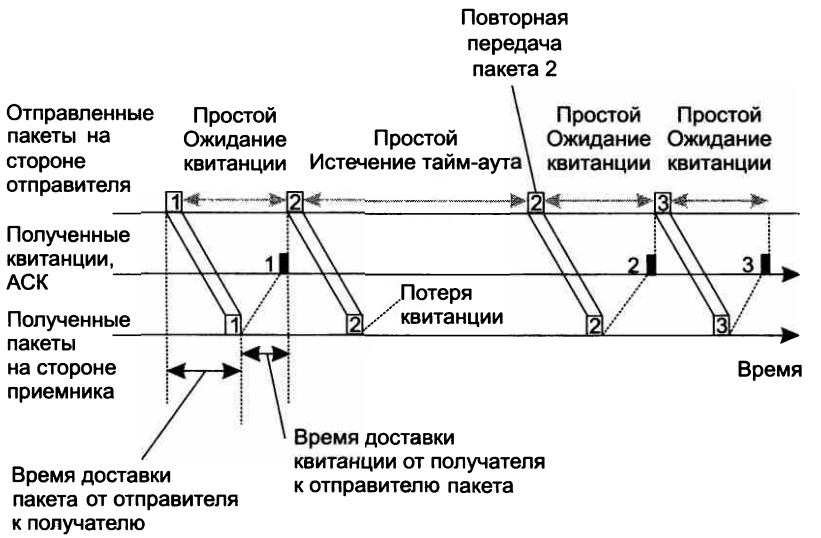


Рис. 16.10. Метод простоя источника

Если первая квитанция не была утеряна, а просто шла к отправителю слишком долго, то возможна коллизия с приходом в источник двух квитанций на пакет-оригинал и пакет-дубль. Вторую квитанцию отправитель может интерпретировать как квитанцию на получение следующего по порядку пакета. Таким образом, отправитель также должен иметь возможность распознавать дубликаты квитанций.

Частичное решение задачи распознавания дубликатов может быть достигнуто за счет включения в заголовок пакета *специального бита*. Этот бит устанавливается отправителем так, что нуль и единица в качестве его значения чередуются в пределах всей последовательности отправляемых пакетов. Приемник проверяет значение бита: если у двух последовательно пришедших пакетов значения данного бита равны двум единицам или двум нулям,

то эти пакеты считаются дубликатами. Аналогично поступает отправитель с квитанциями. Такой способ распознавания дубликатов не является абсолютно надежным, поскольку он основывается на предположении, что вероятность прихода в приемник дубликатов друг за другом достаточно высока.

Очевидно, что в данном методе коэффициент использования линии связи очень низкий — основную часть времени передатчик простаивает в ожидании прихода квитанции.

Концепция скользящего окна

Концепция **скользящего окна** (sliding window) заключается в том, что для повышения скорости передачи данных отправителю разрешается передать некоторое количество пакетов, *не дожидаясь прихода на эти пакеты квитанций*.

Так как в этом методе одновременно могут существовать несколько неподтвержденных пакетов, то их необходимо каким-то образом различать, чтобы отправитель мог понять, получение какого пакета подтверждает квитанция. Для идентификации пакетам присваиваются *уникальные последовательные номера*, которые размещаются в заголовках пакетов. Разрядность поля «номер пакета» определяет диапазон возможных номеров. Когда этот диапазон исчерпывается, нумерация пакетов снова начинается с нуля. Таким образом, нельзя абсолютно исключить ситуацию, когда в сети существуют пакеты с одинаковыми номерами. Позже мы еще вернемся к этому вопросу.

Окно определяется на последовательности пронумерованных пакетов (рис. 16.11). Окно всегда имеет нижнюю границу, называемую также **базой окна** (здесь это пакет с номером 9), и верхнюю границу (14). Количество номеров, попадающих в пределы окна, называют **размером окна** (6). Очевидно, что окно всегда перемещается в сторону больших номеров.

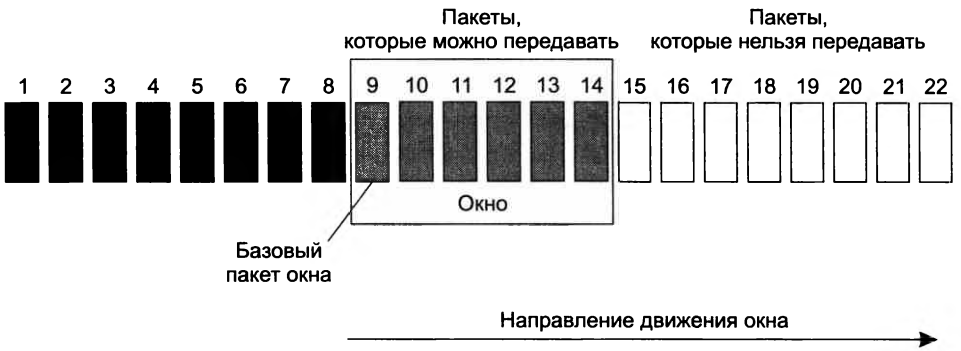


Рис. 16.11. Окно передачи

Окно, показанное на рисунке, регулирует процесс передачи — оно ограничивает количество пакетов, которые отправитель может передать до получения квитанции. Окно может быть определено не только для передачи, но и для приема пакетов, в таком случае ограничивается количество пакетов, которые получатель может принять, не передавая их на верхний уровень.

ПРИМЕЧАНИЕ

Может возникнуть вопрос, зачем вообще нужно это ограничение, почему нельзя разрешить отправителю передавать произвольное количество пакетов? Дело в том, что, сняв ограничение и позволив тем самым существовать в сети большому количеству искаженных, потерявшихся и не получивших подтверждения пакетов, мы сделаем процесс передачи неконтролируемым, в результате, например, для хранения таких пакетов потребуется буфер неограниченных размеров.

На рис. 16.12 показана идеализированная схема «скольжения» окна передачи. В этом примере предполагается, что пакеты и квитанции не теряются и приходят в том же порядке, в каком были отправлены. При этом интервалы между пакетами и квитанциями являются неравномерными. Движение окна определяется, во-первых, поступлением квитанций — подтверждение успешного приема очередного пакета позволяет переместить окно вперед, во-вторых, исчерпанием окна — окно приостанавливается, когда отправитель передал все пакеты из окна, но не получил ни на один из них квитанции.

Здесь последовательность номеров пакетов отображается на оси ординат, а значит, окно скользит вдоль вертикали.

Окно имеет размер 5. В начальный момент, когда еще не послано ни одного пакета, окно определяет диапазон номеров пакетов от 1 до 5 включительно. Источник начинает передавать пакеты и через какое-то время получает в ответ квитанции. Для простоты предпо-

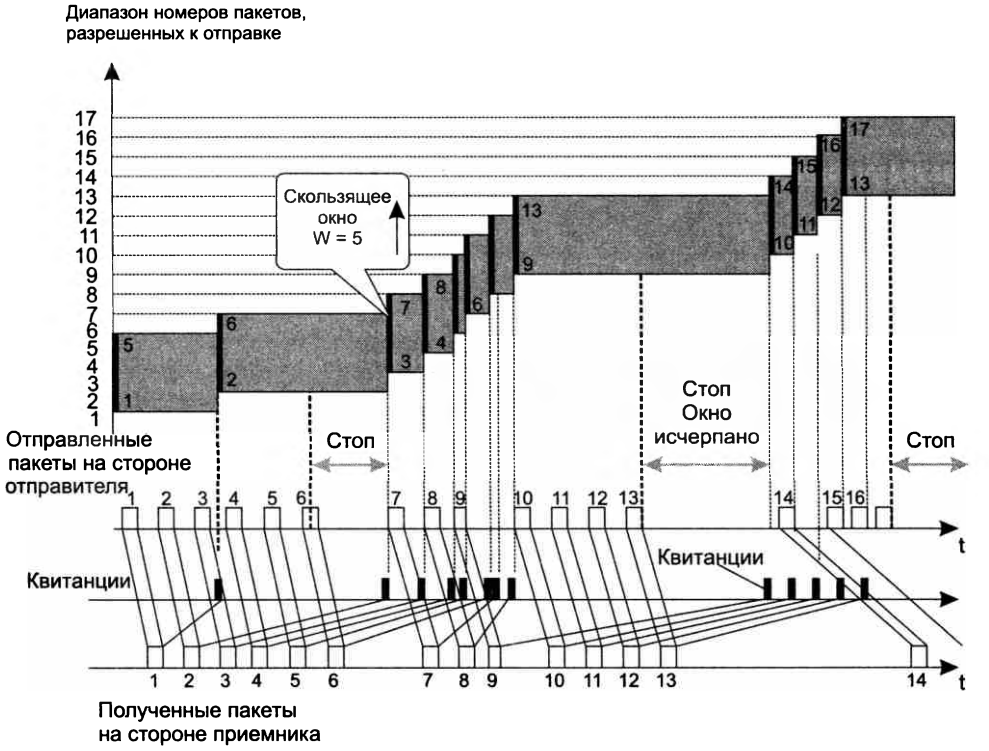


Рис. 16.12. Метод скользящего окна

ложим, что квитанции поступают в той же последовательности (но не обязательно в том же темпе), что и пакеты, которым они соответствуют. В момент получения отправителем квитанции 1 окно сдвигается на одну позицию вверх, определяя новый диапазон разрешенных к отправке пакетов (от 2 до 6).

Процессы отправки пакетов и получения квитанций идут независимо друг от друга. В нашем примере отправитель продолжает передавать пакеты, но некоторое время не получает на них квитанции. После передачи пакета 6 окно исчерпывается, и источник приостанавливает передачу.

После получения квитанции 2 (на пакет 2) окно сдвигается вверх на единицу, определяя диапазон разрешенных к передаче пакетов от 3 до 7. Аналогичное «скольжение» окна вверх происходит после получения каждой квитанции: окно сдвигается вверх на 1, но его размер при этом не меняется. После прихода квитанции 8 окно оказывается в диапазоне от 9 до 13 и остается таковым достаточно долго, так как по каким-то причинам источник перестает получать подтверждения о доставке пакетов. Отправив последний разрешенный пакет 13, передатчик снова прекращает передачу, с тем чтобы возобновить ее после прихода квитанции 9.

Теперь, когда мы познакомились с концепцией скользящего окна, рассмотрим методы передачи, построенные на ее основе.

Передача с возвращением на N пакетов

В данном методе ставится задача обеспечить более эффективное использование линии связи, чем в методе с простоем источника. Для этого отправителю разрешается отправлять следующие пакеты, не дожидаясь подтверждения получателем их успешного приема. Количество переданных таким образом пакетов ограничивается окном.

Поскольку отправитель должен иметь возможность при необходимости повторить передачу любого из переданных пакетов, их необходимо буферизовать. Кроме того, отправитель должен отслеживать статус пакетов (отправлен/не отправлен, подтвержден/не подтвержден).

На рис. 16.13 показано положение окна передачи в некоторый момент времени. Базой окна в данном случае является номер 7, а верхней границей — номер 17. Все пакеты с номерами в этом диапазоне отправитель имеет право передавать независимо от поступления квитанций. С левой стороны окна находятся более «старые» пакеты (1–6), которые были переданы и подтверждены. Пакеты (18–22) справа от окна в данный момент передавать запрещено.

В пределах окна имеются как уже отправленные, но не подтвержденные пакеты (7–12), так и неотправленные пакеты, которые разрешено отправлять (13–17). Как только самый «старый» пакет в окне (7) получит подтверждение успешного приема, окно сдвинется направо на единицу.

Рассмотрим *алгоритм работы получателя*.

При поступлении нового пакета получатель всегда выполняет одни и те же действия, проверяя:

- является ли пакет неискаженным;
- является ли он следующим по порядку в последовательности уже полученных пакетов.

Рис. 16.13. Окно передачи в методе с возвратом на N пакетов

Если оба условия выполнены, то пакет принимается и передается на более высокий уровень (размер окна приема в этом методе равен 1), а отправителю посылается квитанция, в которой указывается номер успешно принятого пакета.

Если хотя бы одно из этих условий не выполнено, то пакет *отбрасывается*, а отправителю снова посылается квитанция с номером последнего по времени успешно принятого пакета.

Отметим очень важный факт: в том случае, когда получатель принимает и подтверждает прием пакетов в строгом соответствии с их порядковыми номерами (а именно с такой ситуацией мы сейчас имеем дело), квитанция о любом принятом пакете говорит также о том, что *все* предыдущие пакеты также были приняты успешно. Такого рода квитанции называют *кумулятивными*, или *накопительными*. Другими словами, нет необходимости дублировать потерянную кумулятивную квитанцию, потому что она компенсируется приходом следующей квитанции, также являющейся кумулятивной.

Теперь взглянем на *алгоритм работы отправителя*.

Отправитель использует *один* таймер, значение тайм-аута которого устанавливается равным предельному времени ожидания квитанции, отправленной получателем для подтверждения успешности доставки и приема *базового пакета* окна.

На процесс передачи пакетов влияют следующие события:

- *Исчерпание окна* — ситуация, когда все пакеты из окна отправлены, но не подтверждены. В этом случае *передача останавливается*.
- *Истечение тайм-аута* — это событие интерпретируется отправителем как потеря пакета или квитанции, а значит, *выполняется повторная передача* базового пакета и для него *заново устанавливается таймер*. При этом, что очень важно, повторно передается не только этот пакет, но и *все* переданные, но не подтвержденные пакеты. Именно такая реакция на недоставленный пакет дала название этому методу «возвращение на N пакетов». Повторная передача всех неподтвержденных пакетов нужна, так как даже если приемник и получил их, он их отбросил, так как они не образовывали непрерывную последовательность пакетов.
- *Поступление квитанции*. Соответствующий пакет и все пакеты в пределах окна с меньшими номерами считаются успешно принятыми (учитывается кумулятивность квитанции). *Окно сдвигается*, фиксируется новый базовый пакет, переустанавливается

таймер. Если до этого передача была остановлена из-за исчерпания окна, то *передача возобновляется*.

Если квитанция на базовый пакет пришла после истечения его тайм-аута, а значит, и после его повторной передачи, эта квитанция «засчитывается», и выполняются все действия, определенные для этого случая.

Зная алгоритм работы отправителя, мы можем легко предсказать, что произойдет с преждевременно пришедшим и поэтому отброшенным пакетом. Есть две возможности. Либо истечет тайм-аут для одного из предшествующих пакетов, и он вместе со всеми остальными будет передан повторно. Либо этот пакет сам станет базовым, и после непременного истечения его тайм-аута (он ведь был отброшен) произойдет его повторная передача.

Поводя итог, отметим, что алгоритм работы получателя в этом методе существенно проще, чем отправителя. Получатель не использует окно, а следовательно, не нуждается в буферизации пакетов и отслеживании их статуса. От получателя требуется только распознавать ошибочные пакеты и отслеживать последовательность их номеров.

Эффективность данного метода выше по сравнению с методом простоя источника за счет передачи в линию связи сразу нескольких пакетов. Однако для него характерна *избыточность*: во-первых, получатель отбрасывает не только искаженный, но и корректно принятый пакет, если его номер выбивается из последовательности, во-вторых, отправитель повторно передает не только потерянный или искаженный пакет, но и все пакеты, которые были отправлены после него.

Передача с выборочным повторением

В данном методе, как это следует из его названия, получатель может *выборочно* запросить повторную передачу отдельного пакета, а не всей последовательности переданных пакетов, как это происходит при передаче с возвращением на N пакетов.

Чтобы избежать избыточных повторных передач, принимающей стороне запрещено отбрасывать правильно принятый пакет лишь потому, что его номер выбивается из последовательности. А это значит, что данный пакет, как и другие нарушающие последовательность пакеты, получатель должен где-то временно сохранять. Для этих целей получатель организует буфер, где он хранит и упорядочивает прибывающие пакеты в соответствии с их номерами.

Согласно поставленной задаче выборочного повторения ошибочных пакетов, выборочными (индивидуальными) должны быть и квитанции, позволяющие подтверждать успешность приема каждого отдельного пакета, и таймеры, замеряющие тайм-аут для каждого отправленного пакета[1].

В данном методе концепция скользящего окна используется как для передачи, так и для приема пакетов. Оба окна определены на одной и той же последовательности номеров пакетов, размеры окон *одинаковы*. Однако их «скольжение» не является синхронным. В зависимости от возникающих ошибок при передаче пакетов и квитанций окно приема может «опережать» окно передачи. Отправитель и получатель работают только с пакетами, находящимися в пределах окна передачи и окна приема соответственно.

---

[1] Обычно в протоколах используется множество программных таймеров, построенных на основе одного аппаратного.

На рис. 16.14 показано положение окон передачи и приема в некоторый момент времени. Отправитель уже передал принятые от верхнего уровня пакеты 1–6, получил на них квитанции и ждет квитанцию на переданный пакет с номером 7. А получатель собрал и передал своему верхнему уровню непрерывную последовательность пакетов 1–8, но все еще не получил пакет 9. Окно передачи в данный момент позиционируется между базовым пакетом 7 и граничным пакетом 17, а окно приема — в границах 9–19. Все пакеты с номерами, попадающими в окна, отправитель имеет право передавать, а получатель — принимать независимо от поступления квитанций.

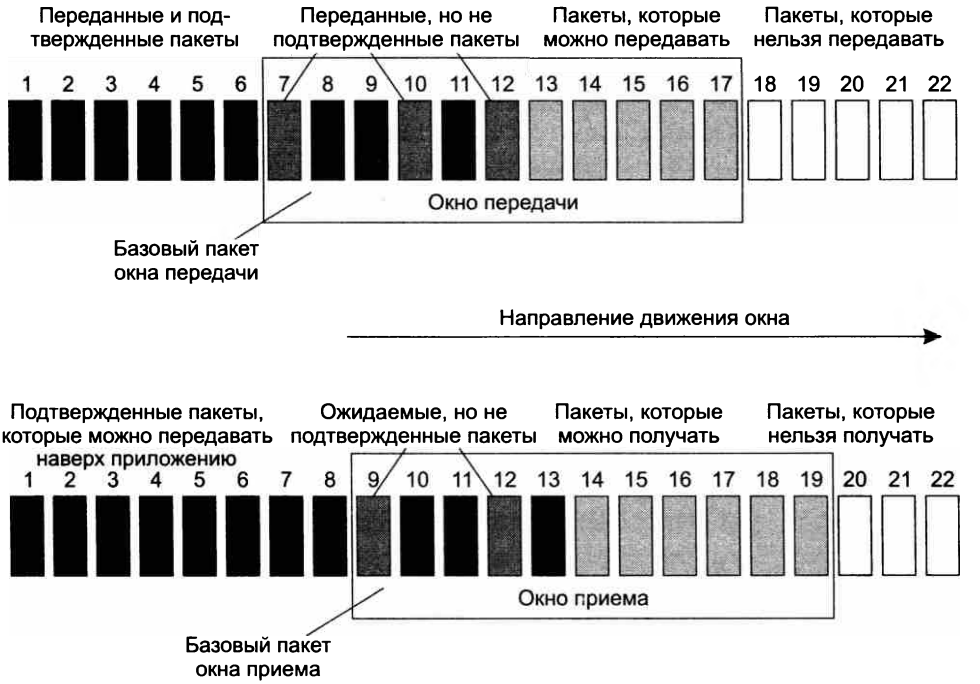


Рис. 16.14. Окна передачи и приема при выборочном повторении

Поскольку в этом методе квитанция является индивидуальной, а не накопительной, то в окне передачи среди переданных пакетов могут быть как подтвержденные (8, 9, 11), так и неподтвержденные (7, 10, 12).

Номера из окна приема в общем случае могут соответствовать:

- принятым и подтвержденным пакетам (10, 11, 13), которые не могут быть переданы из буфера верхнему уровню, так как в их последовательность «вклинились» номера некоторых отсутствующих пакетов (9, 12);
- ожидаемым, еще не полученным пакетам (9, 12);
- пакетам, которые получателю разрешено принимать (14–19), так как их номера попадают в окно приема.

Получатель принимает, размещает в буфере и подтверждает квитанцией любой принятый пакет при условии, что он не искажен и его номер попадает в окно приема. Если принят

базовый пакет, то левая граница окна приема сдвигается до первого ожидаемого, но еще не полученного пакета. На рисунке базовым является пакет 9. Когда он будет успешно принят, пробел с недостающим пакетом заполнится, окно сдвинется, и новым базовым пакетом станет пакет 12. Выдвинувшаяся за границу окна приема непрерывная последовательность пакетов 9, 10, 11 тогда может быть передана верхнему уровню.

На работе отправителя сказываются следующие события:

- *Исчерпание окна.* Отправитель последовательно посылает пакеты до тех пор, пока не исчерпается окно передачи.
- *Приход квитанции.* Отправитель, получив квитанцию, присваивает пакету статус успешно переданного. Если это был базовый пакет (например, пакет 7 на рисунке), то окно смещается вправо до первого по порядку принятого, но не подтвержденного пакета, который становится базой (на рисунке пакет 10).
- *Истечение тайм-аута.* Таймер устанавливается для каждого пакета отдельно, по истечении тайм-аута соответствующий пакет повторяют. Таким образом, пакет повторяют, только если он был потерян или искажен.

Возможна ситуация, в которой пакет благополучно был принят, а квитанция на него потерялась. Тогда к получателю придет дубликат пакета. Получатель не должен его игнорировать, ему следует подтвердить квитанцией прием дубликата, иначе отправитель «застрянет» на этом пакете, бесконечно повторяя его.

Подводя итог, можно отметить, что методы, использующие скользящее окно, сложнее в реализации, чем метод простоя источника, так как в первом случае требуется поддержание буфера (или буферов в случае выборочной передачи), отслеживание номеров и статуса пакетов, а также определение по меньшей мере двух важных параметров алгоритма, таких как размер окна и величина тайм-аута.

Концепция скользящего окна используется во многих протоколах, обеспечивающих надежную передачу данных, например в протоколе TCP, рассматриваемом в следующем разделе, а также в протоколах HDLC и LAP-M, которые будут изучаться в части V, посвященной технологиям глобальных сетей.

Реализация метода скользящего окна в протоколе TCP

Сегменты и поток байтов

Алгоритм скользящего окна в протоколе TCP имеет некоторые существенные особенности. В частности, в рассмотренном обобщенном алгоритме скользящего окна единицей передаваемых данных является пакет, и размер окна также определяется в кадрах, в то время как в протоколе TCP дело обстоит совсем по-другому.

Хотя единицей передаваемых данных протокола TCP является сегмент (аналог кадра в данном контексте), окно определено на множестве нумерованных байтов неструктурированного потока данных, передаваемого приложением протоколу TCP.

В ходе переговорного процесса модули TCP обеих участвующих в обмене сторон договариваются между собой о параметрах процедуры обмена данными. Одни из них остаются постоянными в течение всего сеанса связи, другие в зависимости, например, от интенсивности трафика и/или размеров буферов адаптивно изменяются. Одним из таких параметров является *начальный номер байта*, с которого будет вестись отсчет в течение всего функционирования данного соединения. У каждой стороны свой начальный номер. Нумерация байтов в пределах сегмента осуществляется начиная от заголовка (рис. 16.15).

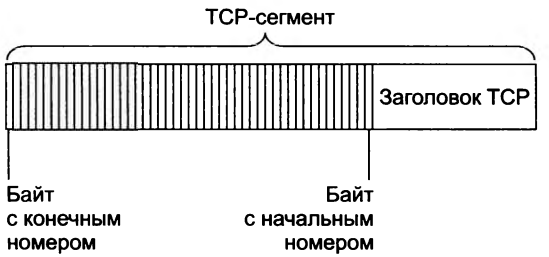


Рис. 16.15. Нумерация байтов в TCP-сегменте

Когда отправитель посылает TCP-сегмент, он помещает в поле *последовательного номера* номер первого байта данного сегмента, который служит *идентификатором* сегмента. На рис. 16.16 показано четыре сегмента размером 1460 байт и один — 870 байт. Идентификаторами этих сегментов являются номера 32600, 34060, 35520 и т. д. На основании этих номеров получатель TCP-сегмента не только отличает данный сегмент от других, но и позиционирует полученный фрагмент относительно общего потока байтов. Кроме того, он может сделать вывод, например, что полученный сегмент является дубликатом или что между двумя полученными сегментами пропущены данные и т. д.

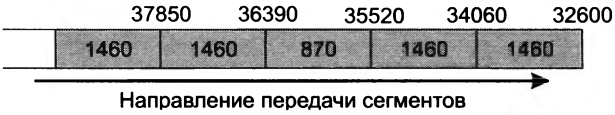


Рис. 16.16. Порядковый номер и номер квитанции

В качестве квитанции получатель сегмента отсылает ответное сообщение (сегмент), в поле *подтвержденного номера* которого он помещает число, на единицу превышающее максимальный номер байта в полученном сегменте. Так, для первого отправленного сегмента, изображенного на рисунке, квитанцией о получении (подтвержденным номером) будет число 34060, для второго — 35520 и т. д. Подтвержденный номер часто интерпретируют не только как оповещение о благополучной доставке, но и как номер следующего ожидаемого байта данных.

Квитанция в протоколе TCP посылается только в случае правильного приема данных. Таким образом, отсутствие квитанции означает либо потерю сегмента, либо потерю квитанции, либо прием искаженного сегмента.

В соответствии с определенным форматом один и тот же TCP-сегмент может нести в себе как пользовательские данные (в поле данных), так и квитанцию (в заголовке), которой подтверждается получение данных от другой стороны.

Система буферов при дуплексной передаче

Поскольку протокол TCP является дуплексным, каждая сторона одновременно выступает и как отправитель, и как получатель. У каждой стороны есть пара буферов: один — для хранения принятых сегментов, другой — для сегментов, которые только еще предстоит отправить. Кроме того, имеется буфер для хранения копий сегментов, которые были отправлены, но квитанции о получении которых еще не поступили (рис. 16.17).

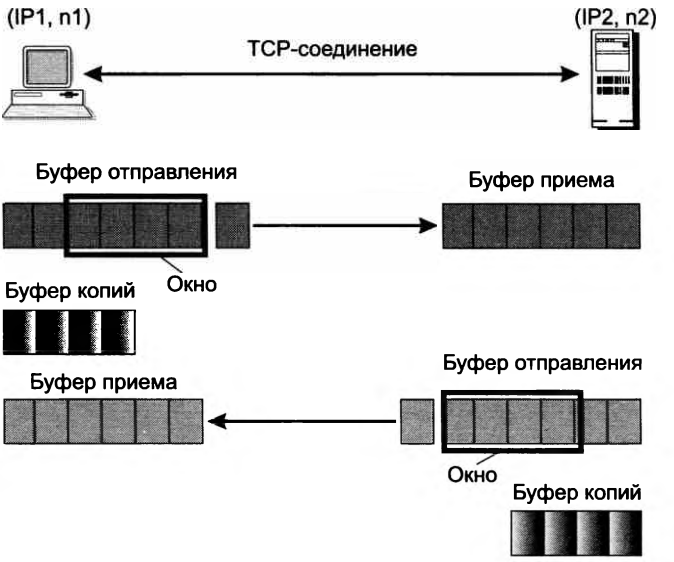


Рис. 16.17. Система буферов TCP-соединения

И при установлении соединения, и в ходе передачи обе стороны, выступая в роли получателя, посылают друг другу **окна приема**. Каждая из сторон, получив окно приема, «узнает», сколько байтов ей разрешается отправить с момента получения последней квитанции. Другими словами, посылая окна приема, обе стороны пытаются регулировать поток байтов в свою сторону, сообщая своему «визави», какое количество байтов (начиная с номера байта, о котором уже была выслана квитанция) они готовы в настоящий момент принять.

На рис. 16.18 показан поток байтов, поступающий от приложения в выходной буфер модуля TCP. Из потока байтов модуль TCP «нарезает» последовательность сегментов и поочередно отправляет их приложению-получателю. Для определенности на рисунке принято направление перемещения данных справа налево. В этом потоке можно указать несколько логических границ:

- Первая граница отделяет сегменты, которые уже были отправлены и на которые уже пришли квитанции. Последняя квитанция пришла на байт с номером *N*.
- По другую сторону этой границы располагается окно размером *W* байт. Часть байтов, входящих в окно, составляют сегменты, которые также уже отправлены, но квитанции на которые пока не получены.
- Оставшаяся часть окна — это сегменты, которые пока не отправлены, но могут быть отправлены, так как входят в пределы окна.

- И наконец, последняя граница указывает на начало последовательности сегментов, ни один из которых не может быть отправлен до тех пор, пока не придет очередная квитанция и окно не будет сдвинуто вправо.

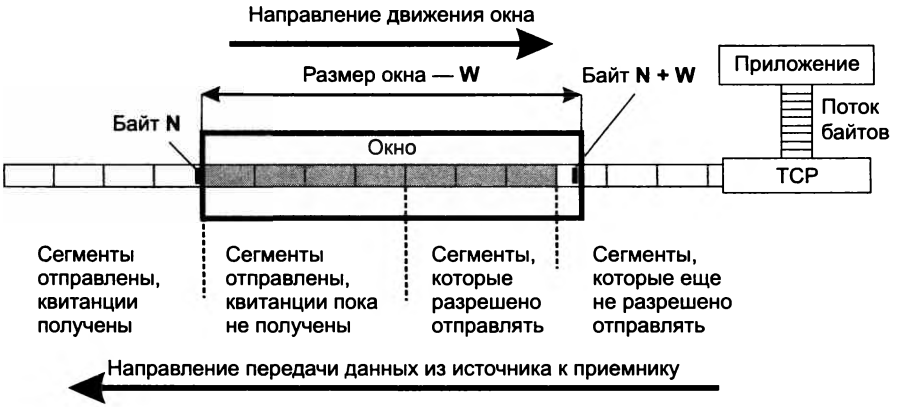


Рис. 16.18. Особенности реализации алгоритма скользящего окна в протоколе TCP

Накопительный принцип квитирования

Если размер окна равен *W*, а последняя по времени квитанция содержала значение *N*, то отправитель может посылать новые сегменты до тех пор, пока в очередной сегмент не попадет байт с номером *N* + *W*. Этот сегмент выходит за рамки окна, и передачу в таком случае необходимо приостановить до прихода следующей квитанции.

Получатель может послать квитанцию, подтверждающую получение сразу нескольких сегментов, если они образуют непрерывный поток байтов. Например (рис. 16.19, *а*), если в буфер, плотно, без пропусков заполненный потоком байтов до 2354 включительно, поочередно поступили сегменты (2355–3816), (3817–5275) и (5276–8400), где цифры в скобках обозначают номера первых и последних байтов каждого сегмента, то получателю достаточно отправить только одну квитанцию на все три сегмента, указав в ней в качестве номера квитанции значение 8401. Таким образом, процесс квитирования в TCP является *накопительным*.

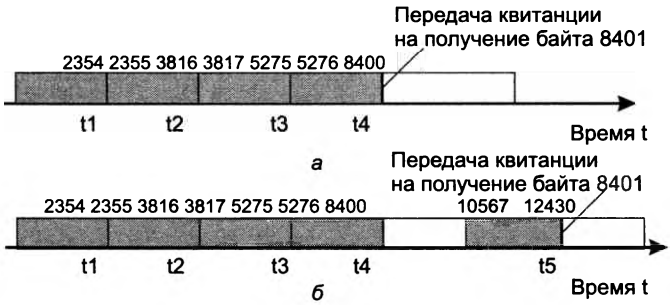


Рис. 16.19. Накопительный принцип квитирования: *а* — плотное заполнение буфера (в момент t4 передается квитанция на байт 8401), *б* — неплотное заполнение буфера (в момент t5 снова передается квитанция на байт 8401)

Вполне возможны ситуации, когда сегменты приходят к получателю не в том порядке, в каком были посланы, то есть в приемном буфере может образоваться «прогалина» (рис. 16.19, *б*). Пусть, к примеру, после указанных ранее трех сегментов вместо следующего по порядку сегмента (8401–10566) пришел сегмент (10567–12430). Очевидно, что послать в качестве номера квитанции значение 12431 нельзя, потому что это бы означало, что получены все байты вплоть до 12430. Поскольку в потоке байтов образовался разрыв, получатель может только еще раз повторить квитанцию 8401, говоря тем самым, что все еще ожидает поступления потока байтов, начиная с 8401, то есть подтверждает получение не отдельных блоков данных, а непрерывной последовательности байтов.

Когда протокол TCP передает в сеть сегмент, он «на всякий случай» помещает его копию в буфер, называемый также очередью повторной передачи, и запускает таймер. Когда приходит квитанция на этот сегмент, соответствующая копия удаляется из очереди. Если же квитанция не приходит до истечения срока, то сегмент, вернее его копия, посылается повторно. Может случиться так, что копия сегмента придет тогда, когда исходный сегмент уже окажется на месте, тогда дубликат попросту отбрасывается.

Параметры управления потоком в TCP

Какой размер окна должен назначить источник приемнику, и наоборот? Точнее, каким на каждой из сторон должно быть выбрано время ожидания (тайм-аут) очередной квитанции? От ответа на этот вопрос зависит производительность протокола TCP.

При выборе величины *тайм-аута* должны учитываться скорость и надежность линий связи, их протяженность и многие другие факторы. Тайм-аут не должен быть слишком коротким, чтобы по возможности исключить избыточные повторные передачи, снижающие полезную пропускную способность системы, но он не должен быть и слишком длинным, чтобы избежать длительных простоев, связанных с ожиданием несуществующей или «заблудившейся» квитанции.

В протоколе TCP тайм-аут определяется с помощью достаточно сложного *адаптивного* алгоритма, идея которого состоит в следующем. При каждой передаче засекается время от момента отправки сегмента до прихода квитанции о его приеме (время оборота). Получаемые значения времени оборота усредняются с весовыми коэффициентами, возрастающими от предыдущего замера к последующему. Это делается с тем, чтобы усилить влияние последних замеров. В качестве тайм-аута выбирается среднее время оборота, умноженное на некоторый коэффициент. Практика показывает, что значение этого коэффициента должно превышать 2. В сетях с большим разбросом времени оборота при выборе тайм-аута учитывается также дисперсия этой величины.

Размер окна приема связан с наличием в данный момент места в буфере данных у принимающей стороны. Поэтому в общем случае окна приема на разных концах соединения имеют разный размер. Например, можно ожидать, что сервер, вероятно обладающий большим буфером, пошлет клиентской станции окно приема большее, чем клиент серверу. В зависимости от состояния сети то одна, то другая стороны могут объявлять новые значения окон приема, динамически уменьшая и увеличивая их.

Варьируя величину окна, можно влиять на загрузку сети. Чем больше окно, тем бóльшая порция неподтвержденных данных может быть послана в сеть. Но если пришло бóльшее количество данных, чем может быть принято модулем TCP, данные отбрасываются.

Это ведет к излишним пересылкам информации и ненужному росту нагрузки на сеть и модуль TCP.

В то же время окно малого размера может ограничить передачу данных скоростью, которая определяется временем путешествия по сети каждого посылаемого сегмента. Чтобы избежать применения малых окон, в некоторых реализациях TCP получателю данных предлагается откладывать реальное изменение размеров окна до тех пор, пока свободное место не составит 20–40 % от максимально возможного объема памяти для этого соединения. Но и отправителю не стоит спешить с посылкой данных, пока окно принимающей стороны не станет достаточно большим. Учитывая эти соображения, разработчики протокола TCP предложили схему, согласно которой при установлении соединения заявляется большое окно, но впоследствии его размер существенно уменьшается. Существуют и другие прямо противоположные алгоритмы настройки окна, когда вначале выбирается минимальное окно, а затем, если сеть справляется с предложенной нагрузкой, его размер резко увеличивается.

Управлять размером окна приема может не только та сторона, которая посылает это окно, чтобы регулировать поток данных в свою сторону, но и вторая сторона — потенциальный отправитель данных. Если вторая сторона фиксирует ненадежную работу линии связи (регулярно запаздывают квитанции, часто требуется повторная передача), то она может по собственной инициативе уменьшить окно. В таких случаях действует правило: в качестве действующего размера окна выбирается минимальное из двух значений: значения, диктуемого приемной стороной, и значения, определяемого «на месте» отправителем.

Признаком перегрузки TCP-соединения является возникновение очередей на промежуточных узлах (маршрутизаторах) и на конечных узлах (компьютерах). При переполнении приемного буфера конечного узла «перегруженный» модуль TCP, отправляя квитанцию, помещает в нее новый уменьшенный размер окна. Если он совсем отказывается от приема, то в квитанции указывается *окно нулевого размера*. Однако даже после этого приложение может послать сообщение на отказавшийся от приема порт. Для этого сообщение должно сопровождаться *указателем срочности*. В такой ситуации порт обязан принять сегмент, даже если для этого придется вытеснить из буфера уже находящиеся там данные. После приема квитанции с нулевым значением окна протокол-отправитель время от времени делает контрольные попытки продолжить обмен данными. Если протокол-приемник уже готов принимать информацию, то в ответ на контрольный запрос он посылает квитанцию с указанием ненулевого размера окна.

Как видно из нашего далеко не полного описания двух протоколов транспортного уровня стека TCP/IP, на один из них — TCP — возложена сложная и очень важная задача обеспечения надежной передачи данных через ненадежную сеть.

В то же время функциональная простота протокола UDP обусловливает простоту алгоритма его работы, компактность и быстродействие. Поэтому те приложения, в которых реализован собственный достаточно надежный механизм обмена сообщениями, основанный на установлении соединения, предпочитают для непосредственной передачи данных по сети использовать менее надежные, но более быстрые средства транспортировки, в качестве которых по отношению к протоколу TCP и выступает протокол UDP. Протокол UDP может применяться и тогда, когда хорошее качество линий связи обеспечивает достаточный уровень надежности и без применения дополнительных приемов наподобие установления логического соединения и квитирования передаваемых пакетов. Заметим также, что по-

скольку протокол TCP основан на логических соединениях, он, в отличие от протокола UDP, *не годится для широковещательной и групповой рассылки.*

Выводы

В то время как задачей протокола IP является передача данных между сетевыми интерфейсами в составной сети, основная задача протоколов TCP и UDP заключается в передаче данных между прикладными процессами, выполняющимися на разных конечных узлах сети.

Протокол UDP является дейтаграммным протоколом, работающим без установления логического соединения, он не гарантирует доставку своих сообщений, а следовательно, не компенсирует ненадежность дейтаграммного протокола IP.

Системные очереди к точкам входа прикладных процессов называют портами. Порты идентифицируются номерами и однозначно определяют приложение в пределах компьютера. Если процессы представляют собой популярные общедоступные службы, такие как FTP, telnet, HTTP, TFTP, DNS и т. п., то за ними централизованно закрепляются стандартные (назначенные) номера.

TCP решает задачу надежного обмена данными путем установления логических соединений. Соединение однозначно идентифицируется парой сокетов.

Сокетом прикладного процесса называется пара из IP-адреса и номера порта.

Для организации надежного обмена данными применяются методы передачи с квитированием, подразделяемые на методы простоя источника и методы скользящего окна.

Для управления потоком в рамках TCP-соединения используется специфический вариант алгоритма скользящего окна. Сторона-получатель передает стороне-отравителю размер окна приема в байтах.

Контрольные вопросы

1. Система DNS может использовать для доставки своих сообщений как протокол UDP, так и протокол TCP. Какой вариант вы считаете более предпочтительным? Аргументируйте свой ответ.
2. Если при обмене данными по методу с возвращением на N пакетов отправитель получил квитанцию на (n+1)-й пакет, а квитанция на предыдущий n-й пакет не пришла, то:

   а) после истечения тайм-аута отправитель повторно отсылает n-й пакет;

   б) после истечения тайм-аута получатель повторно отсылает квитанцию на n-й пакет;

   в) отправитель считает n-й пакет успешно принятым и продолжает передачу.
3. Как соотносятся размеры окна приема и окна передачи в методе с выборочным повторением? Поясните свой ответ. Варианты ответов:

   а) они равны;

   б) окно передачи больше, чем окно приема;

   в) окно приема больше, чем окно передачи.

4. Какой объем данных получен в течение TCP-сеанса отправителем TCP-сегмента, в заголовке которого в поле квитанции помещено значение 180005? Известно, что первый полученный байт имел номер 15000.
5. В каком виде передаются квитанции на получение сегментов в протоколе TCP? Варианты ответов:
   а) квитанция передается в поле данных TCP-сегмента, с установленным в заголовке флагом ACK:
   б) квитанция — это значение поля подтвержденного номера в заголовке TCP-сегмента с установленным флагом ACK;
   в) квитанция — это значение поля последовательного номера в заголовке TCP-сегмента с установленным флагом ACK;
   г) квитанция — это флаг ACK.
6. Как влияет на эффективность передачи протокола TCP размер окна? Величина таймаута?

ГЛАВА 17 Протоколы маршрутизации

Общие свойства и классификация протоколов маршрутизации

Протоколы маршрутизации обеспечивают поиск и фиксацию маршрутов продвижения данных через составную сеть TCP/IP. Давайте остановимся на некоторых общих свойствах протоколов данного класса.

Начнем с упоминания о таких способах продвижения пакетов в составных сетях, которые вообще *не требуют наличия таблиц маршрутизации на маршрутизаторах*.

Наиболее простым способом передачи пакетов по сети является так называемая **лавинная маршрутизация**, когда каждый маршрутизатор передает пакет всем своим непосредственным соседям, исключая тот, от которого его получил. Понятно, что это — не самый рациональный способ, так как пропускная способность сети используется крайне расточительно, тем не менее такой подход работоспособен (именно так мосты и коммутаторы локальных сетей поступают с кадрами, имеющими неизвестные адреса).

Еще одним видом маршрутизации, не требующим наличия таблиц маршрутизации, является **маршрутизация от источника** (source routing). В этом случае отправитель помещает в пакет информацию о том, какие промежуточные маршрутизаторы должны участвовать в передаче пакета к сети назначения. На основе этой информации каждый маршрутизатор считывает адрес следующего маршрутизатора, и если он действительно является адресом его непосредственного соседа, передает ему пакет для дальнейшей обработки. Вопрос о том, как отправитель узнает точный маршрут следования пакета через сеть, остается открытым. Маршрут может задавать либо вручную администратор, либо автоматически узел-отправитель, но в этом случае ему нужно поддерживать какой-либо протокол маршрутизации, который сообщит ему о топологии и состоянии сети. Маршрутизация от источника была опробована на этапе зарождения Интернета и сохранилась как практически неиспользуемая возможность протокола IPv4. В IPv6 маршрутизация от источника является одним из стандартных режимов продвижения пакетов, существует даже специальный заголовок для реализации этого режима.

Тем не менее большинство протоколов маршрутизации нацелено на *создание таблиц маршрутизации*.

Выбор рационального маршрута может осуществляться на основании различных *критериев*. Сегодня в IP-сетях применяются протоколы маршрутизации, в которых маршрут выбирается по критерию кратчайшего расстояния. При этом расстояние измеряется в различных метриках. Чаще всего используется простейшая метрика — количество хопов, то есть количество маршрутизаторов, которые нужно преодолеть пакету до сети назначения.

В качестве метрик применяются также пропускная способность и надежность каналов, вносимые ими задержки и любые комбинации этих метрик.

Различные протоколы маршрутизации обладают *разным временем конвергенции*.

Протокол маршрутизации должен обеспечить создание на маршрутизаторах *согласованных* друг с другом таблиц маршрутизации, то есть таких таблиц, которые обеспечат доставку пакета от исходной сети в сеть назначения за конечное число шагов. Современные протоколы маршрутизации поддерживают согласованность таблиц, однако это их свойство не абсолютно: при изменениях в сети, например при отказе каналов передачи данных или самих маршрутизаторов, возникают периоды нестабильной работы сети, вызванной временной несогласованностью таблиц разных маршрутизаторов. Протоколу маршрутизации обычно нужно некоторое время, называемое **временем конвергенции**, чтобы после нескольких итераций обмена служебной информацией все маршрутизаторы сети внесли изменения в свои таблицы и в результате таблицы снова стали согласованными.

Различают протоколы, выполняющие статическую и адаптивную (динамическую) маршрутизацию.

При **статической маршрутизации** все записи в таблице имеют неизменяемый, статический статус, что подразумевает бесконечный срок их жизни. Записи о маршрутах составляются и вводятся в память каждого маршрутизатора *вручную администратором сети*. При изменении состояния сети администратору необходимо срочно отразить эти изменения в соответствующих таблицах маршрутизации, иначе может произойти их рассогласование, и сеть будет работать некорректно.

При **адаптивной маршрутизации** все изменения конфигурации сети *автоматически* отражаются в таблицах маршрутизации благодаря *протоколам маршрутизации*. Эти протоколы собирают информацию о топологии связей в сети, что позволяет им оперативно отражать все текущие изменения. В таблицах маршрутизации при адаптивной маршрутизации обычно имеется информация об интервале времени, в течение которого данный маршрут будет оставаться действительным. Это время называют *временем жизни* (TTL) маршрута. Если по истечении времени жизни существование маршрута не подтверждается протоколом маршрутизации, то он считается нерабочим и пакеты по нему больше не посылаются.

Протоколы адаптивной маршрутизации бывают распределенными и централизованными.

При *распределенном* подходе все маршрутизаторы сети находятся в равных условиях, они находят маршруты и строят собственные таблицы маршрутизации, работая в тесной кооперации друг с другом, постоянно обмениваясь информацией о конфигурации сети. При *централизованном* подходе в сети существует один выделенный маршрутизатор, который собирает всю информацию о топологии и состоянии сети от других маршрутизаторов. На основании этих данных выделенный маршрутизатор (который иногда называют *сервером маршрутов*) строит таблицы маршрутизации для всех остальных маршрутизаторов сети, а затем распространяет их по сети, чтобы каждый маршрутизатор получил собственную таблицу и в дальнейшем самостоятельно принимал решение о продвижении каждого пакета.

Применяемые сегодня в IP-сетях протоколы маршрутизации относятся к *адаптивным распределенным* протоколам, которые в свою очередь делятся на две группы:

- дистанционно-векторные алгоритмы (Distance Vector Algorithm, DVA);
- алгоритмы состояния связей (Link State Algorithm, LSA).

В **дистанционно-векторных алгоритмах** (DVA) каждый маршрутизатор *периодически* и *широковещательно* рассылает по сети вектор, компонентами которого являются расстояния (измеренные в той или иной метрике) от данного маршрутизатора до всех известных ему сетей. Пакеты протоколов маршрутизации обычно называют *объявлениями о расстояниях*, так как с их помощью маршрутизатор объявляет остальным маршрутизаторам известные ему сведения о конфигурации сети.

Получив от некоторого соседа вектор расстояний (дистанций) до известных тому сетей, маршрутизатор наращивает компоненты вектора на величину расстояния от себя до данного соседа. Кроме того, он дополняет вектор информацией об известных ему самому других сетях, о которых он узнал непосредственно (если они подключены к его портам) или из аналогичных объявлений других маршрутизаторов. Обновленное значение вектора маршрутизатор рассылает своим соседям. В конце концов каждый маршрутизатор узнает через соседние маршрутизаторы информацию обо всех имеющихся в составной сети сетях и о расстояниях до них.

Затем он выбирает из нескольких альтернативных маршрутов к каждой сети тот маршрут, который обладает наименьшим значением метрики. Маршрутизатор, передавший информацию о данном маршруте, отмечается в таблице маршрутизации как *следующий* (next hop).

Дистанционно-векторные алгоритмы хорошо работают только в небольших сетях. В больших сетях они периодически засоряют линии связи интенсивным трафиком, к тому же изменения конфигурации не всегда корректно могут отражаться алгоритмом этого типа, так как маршрутизаторы не имеют точного представления о топологии связей в сети, а располагают только косвенной информацией — вектором расстояний.

Наиболее распространенным протоколом, основанным на дистанционно-векторном алгоритме, является протокол RIP (см. далее).

Алгоритмы состояния связей (LSA) обеспечивают каждый маршрутизатор информацией, достаточной для построения точного графа связей сети. Все маршрутизаторы работают на основании одного и того же графа, что делает процесс маршрутизации более устойчивым к изменениям конфигурации.

Каждый маршрутизатор использует граф сети для нахождения оптимальных по некоторому критерию маршрутов до каждой из сетей, входящих в составную сеть.

Чтобы понять, в каком состоянии находятся линии связи, подключенные к его портам, маршрутизатор периодически обменивается короткими пакетами HELLO со своими ближайшими соседями. В отличие от протоколов DVA, которые регулярно передают вектор расстояний, протоколы LSA ограничиваются короткими сообщениями, а передача более объемных сообщений происходит только в тех случаях, когда с помощью сообщений HELLO был установлен факт изменения состояния какой-либо связи.

В результате служебный трафик, создаваемый протоколами LSA, гораздо менее интенсивный, чем у протоколов DVA.

Протоколами, основанными на алгоритме состояния связей, являются протокол IS-IS стека OSI (этот протокол используется также в стеке TCP/IP) и протокол OSPF стека TCP/IP.

Протокол RIP

Протокол RIP (Routing Information Protocol — протокол маршрутной информации) является внутренним протоколом маршрутизации дистанционно-векторного типа.

Будучи простым в реализации, этот протокол чаще всего используется в небольших сетях. Для IP имеются две версии RIP — RIPv1 и RIPv2. Протокол RIPv1 не поддерживает масок. Протокол RIPv2 передает информацию о масках сетей, поэтому он в большей степени соответствует требованиям сегодняшнего дня. Так как построение таблиц маршрутизации в обеих версиях протокола принципиально не отличается, в дальнейшем для упрощения записей описывается работа версии 1.

Построение таблицы маршрутизации

Для измерения расстояния до сети стандарты протокола RIP допускают различные виды метрик: хопы, значения пропускной способности, вносимые задержки, надежность сетей (то есть соответствующие признакам D, T и R в поле качества сервиса IP-пакета), а также любые комбинации этих метрик. Метрика должна обладать свойством *аддитивности* — метрика составного пути должна быть равна сумме метрик составляющих этого пути. В большинстве реализаций RIP используется простейшая метрика — количество хопов, то есть количество промежуточных маршрутизаторов, которые нужно преодолеть пакету до сети назначения.

Рассмотрим процесс построения таблицы маршрутизации с помощью протокола RIP на примере составной сети, изображенной на рис. 17.1. Мы разделим этот процесс на 5 этапов.

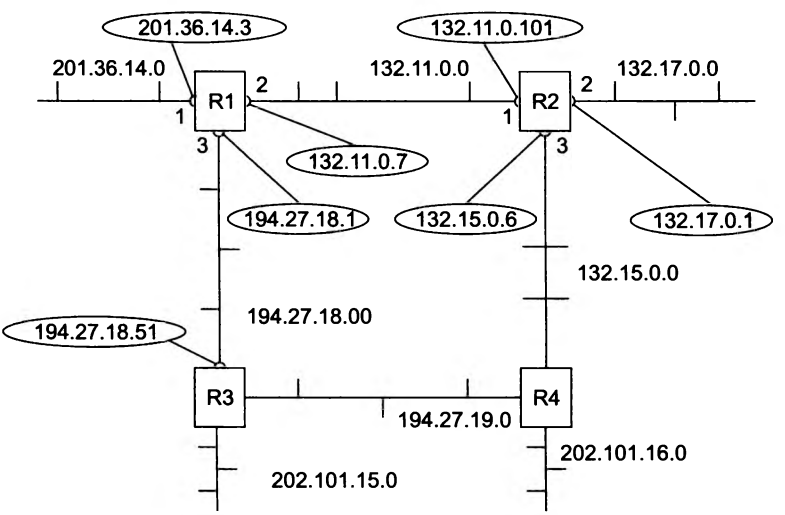


Рис. 17.1. Сеть, построенная на маршрутизаторах RIP

Этап 1 — создание минимальной таблицы. Данная составная сеть включает восемь IP-сетей, связанных четырьмя маршрутизаторами с идентификаторами: R1, R2, R3 и R4.

Маршрутизаторы, работающие по протоколу RIP, могут иметь идентификаторы, однако для протокола они не являются необходимыми. В RIP-сообщениях эти идентификаторы не передаются.

В исходном состоянии на каждом маршрутизаторе программным обеспечением стека TCP/IP автоматически создается минимальная таблица маршрутизации, в которой учитываются только непосредственно подсоединенные сети. На рисунке адреса портов маршрутизаторов в отличие от адресов сетей помещены в овалы.

Таблица 17.1 позволяет оценить примерный вид минимальной таблицы маршрутизации маршрутизатора R1.

Таблица 17.1. Минимальная таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1

Минимальные таблицы маршрутизации в других маршрутизаторах выглядят соответственно, например таблица маршрутизатора R2 состоит из трех записей (табл. 17.2).

Таблица 17.2. Минимальная таблица маршрутизации маршрутизатора R2

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
132.11.0.0	132.11.0.101	1	1
132.17.0.0	132.17.0.1	2	1
132.15.0.0	132.15.0.6	3	1

Этап 2 — рассылка минимальной таблицы соседям. После инициализации каждый маршрутизатор начинает посылать своим соседям сообщения протокола RIP, в которых содержится его минимальная таблица. RIP-сообщения передаются в дейтаграммах протокола UDP и включают два параметра для каждой сети: ее IP-адрес и расстояние до нее от передающего сообщение маршрутизатора.

По отношению к любому маршрутизатору соседями являются те маршрутизаторы, которым данный маршрутизатор может передать IP-пакет по какой-либо своей сети, не пользуясь услугами промежуточных маршрутизаторов. Например, для маршрутизатора R1 соседями являются маршрутизаторы R2 и R3, а для маршрутизатора R4 — маршрутизаторы R2 и R3.

Таким образом, маршрутизатор R1 передает маршрутизаторам R2 и R3 следующие сообщения:

- сеть 201.36.14.0, расстояние 1;
- сеть 132.11.0.0, расстояние 1;
- сеть 194.27.18.0, расстояние 1.

Этап 3 — получение RIP-сообщений от соседей и обработка полученной информации. После получения аналогичных сообщений от маршрутизаторов R2 и R3 маршрутизатор R1 наращивает каждое полученное поле метрики на единицу и запоминает, через какой порт и от какого маршрутизатора получена новая информация (адрес этого маршрутизатора станет адресом следующего маршрутизатора, если эта запись будет внесена в таблицу маршрутизации). Затем маршрутизатор начинает сравнивать новую информацию с той, которая хранится в его таблице маршрутизации (табл. 17.3).

Таблица 17.3. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
194.27.19.0	194.27.18.51	3	2
202.101.15.0	194.27.18.51	3	2
~~132.11.0.0~~	~~132.11.0.101~~	~~2~~	~~2~~
~~194.27.18.0~~	~~194.27.18.51~~	~~3~~	~~2~~

Записи с четвертой по девятую получены от соседних маршрутизаторов, и они претендуют на помещение в таблицу. Однако только записи с четвертой по седьмую попадают в таблицу, а записи восьмая и девятая — нет. Это происходит потому, что они содержат данные об уже имеющихся в таблице маршрутизатора R1 сетях, а расстояние до них больше, чем в существующих записях.

Протокол RIP замещает запись о какой-либо сети только в том случае, если новая информация имеет лучшую метрику (с меньшим расстоянием в хопах), чем имеющаяся. В результате в таблице маршрутизации о каждой сети остается только одна запись; если же имеется несколько записей, равнозначных в отношении путей к одной и той же сети, то все равно в таблице остается одна запись, которая пришла в маршрутизатор первая по времени. Для этого правила существует исключение: если худшая информация о какой-либо сети пришла от того же маршрутизатора, на основании сообщения которого была создана данная запись, то худшая информация замещает лучшую.

Аналогичные операции с новой информацией выполняют и остальные маршрутизаторы сети.

Этап 4 — рассылка новой таблицы соседям. Каждый маршрутизатор отсылает новое RIP-сообщение всем своим соседям. В этом сообщении он помещает данные обо всех известных ему сетях: как непосредственно подключенных, так и удаленных, о которых маршрутизатор узнал из RIP-сообщений.

Этап 5 — получение RIP-сообщений от соседей и обработка полученной информации. Этап 5 повторяет этап 3 — маршрутизаторы принимают RIP-сообщения, обрабатывают содержащуюся в них информацию и на ее основании корректируют свои таблицы маршрутизации. Посмотрим, как это делает маршрутизатор R1 (табл. 17.4).

Таблица 17.4. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
~~132.15.0.0~~	~~194.27.18.51~~	~~3~~	~~3~~
194.27.19.0	194.27.18.51	3	2
~~194.27.19.0~~	~~132.11.0.101~~	~~2~~	~~3~~
~~202.101.15.0~~	~~194.27.18.51~~	~~3~~	~~2~~
202.101.16.0	132.11.0.101	2	3
~~202.101.16.0~~	~~194.27.18.51~~	~~3~~	~~3~~

На этом этапе маршрутизатор R1 получает от маршрутизатора R3 информацию о сети 132.15.0.0, которую тот, в свою очередь, на предыдущем цикле работы получил от маршрутизатора R4. Маршрутизатор уже знает о сети 132.15.0.0, причем старая информация имеет лучшую метрику, чем новая, поэтому новая информация об этой сети отбрасывается.

О сети 202.101.16.0 маршрутизатор R1 узнает на этом этапе впервые, причем данные о ней приходят от двух соседей — от R3 и R4. Поскольку метрики в этих сообщениях указаны одинаковые, то в таблицу попадают данные, пришедшие первыми. В нашем примере считается, что маршрутизатор R2 опередил маршрутизатор R3 и первым переслал свое RIP-сообщение маршрутизатору R1.

Если маршрутизаторы периодически повторяют этапы рассылки и обработки RIP-сообщений, то за конечное время в сети установится корректный режим маршрутизации. Под корректным режимом маршрутизации здесь понимается такое состояние таблиц маршрутизации, когда все сети достижимы из любой сети с помощью некоторого рационального маршрута. Пакеты будут доходить до адресатов и не зацикливаться в петлях, подобных той, которая образуется на рис. 17.1, маршрутизаторами R1, R2, R3 и R4.

Очевидно, если в сети все маршрутизаторы, их интерфейсы и соединяющие их линии связи остаются работоспособными, то объявления по протоколу RIP можно делать достаточно редко, например один раз в день. Однако в сетях постоянно происходят изменения — меняется работоспособность маршрутизаторов и линий связи, кроме того, маршрутизаторы и линии связи могут добавляться в существующую сеть или же выводиться из ее состава.

Для адаптации к изменениям в сети протокол RIP использует ряд механизмов.

Адаптация маршрутизаторов RIP к изменениям состояния сети

К новым маршрутам маршрутизаторы RIP приспосабливаются просто — они передают новую информацию в очередном сообщении своим соседям, и постепенно эта информация становится известна всем маршрутизаторам сети. А вот к изменениям, связанным с потерей какого-либо маршрута, маршрутизаторы RIP адаптируются сложнее. Это связано с тем, что в формате сообщений протокола RIP нет поля, которое бы указывало на то, что путь к данной сети больше не существует.

Для уведомления о том, что некоторый маршрут недействителен, используются два механизма:

- истечение времени жизни маршрута;
- указание специального (бесконечного) расстояния до сети, ставшей недоступной.

Механизм **истечения времени жизни маршрута** основан на том, что каждая запись таблицы маршрутизации (как и записи таблицы продвижения моста/коммутатора), полученная по протоколу RIP, имеет время жизни (TTL). При поступлении очередного RIP-сообщения, которое подтверждает справедливость данной записи, таймер времени жизни устанавливается в исходное состояние, а затем из него каждую секунду вычитается единица. Если за время тайм-аута не придет новое сообщение об этом маршруте, он помечается как *недействительный*.

Время тайм-аута связано с периодом рассылки векторов по сети. В протоколе RIP период рассылки выбран равным 30 секундам, а тайм-аут — шестикратному значению периода рассылки, то есть 180 секундам. Шестикратный запас времени нужен для уверенности в том, что сеть действительно стала недоступной, а не просто произошли потери RIP-сообщений (а это возможно, так как протокол RIP использует транспортный протокол UDP, который не обеспечивает надежной доставки сообщений). Если какой-либо маршрутизатор перестает корректно работать и слать своим соседям сообщения о достижимых через него сетях, то через 180 секунд все записи, порожденные этим маршрутизатором, у его ближайших соседей станут недействительными. После этого процесс повторится уже для соседей ближайших соседей — они вычеркнут подобные записи уже через 360 секунд.

Как видно, сведения о сетях, пути к которым не могут проходить через отказавший маршрутизатор, распространяются по сети не очень быстро. В этом заключается одна из причин выбора в качестве периода рассылки небольшой величины в 30 секунд. Механизм **тайм-аута** работает в тех случаях, когда маршрутизатор не может послать соседям сообщение об отказавшем маршруте, так как либо сам неработоспособен, либо неработоспособна линия связи, по которой можно было бы передать сообщение.

Когда же сообщение послать можно, маршрутизаторы RIP используют прием, заключающийся в *указании бесконечного расстояния до сети, ставшей недоступной*. В протоколе RIP бесконечным условно считается расстояние в 16 хопов. Получив сообщение, в котором расстояние до некоторой сети равно 16 (или 15, что приводит к тому же результату, так как маршрутизатор наращивает полученное значение на 1), маршрутизатор должен проверить, исходит ли эта «плохая» информация о сети от того же маршрутизатора, сообщение которого послужило в свое время основанием для записи о данной сети в таблице маршрутизации. Если это тот же маршрутизатор, то информация считается достоверной и маршрут помечается как недоступный.

Причиной выбора в качестве «бесконечного» расстояния столь небольшого числа является то, что в некоторых случаях отказы связей в сети вызывают длительные периоды некорректной работы маршрутизаторов RIP, выражающейся в зацикливании пакетов в петлях сети. И чем меньше расстояние, используемое в качестве «бесконечного», тем такие периоды короче.

Пример зацикливания пакетов

Рассмотрим случай зацикливания пакетов на примере сети, изображенной на рис. 17.1. Пусть маршрутизатор R1 обнаружил, что его связь с непосредственно подключенной сетью 201.36.14.0 потеряна (например, по причине отказа интерфейса 201.36.14.3). Маршрутизатор R1 отмечает в своей таблице маршрутизации, что сеть 201.36.14.0 недоступна. В худшем случае он обнаружит это сразу же после отправки очередных RIP-сообщений, так что до начала нового цикла его объявлений, в котором он должен сообщить соседям, что расстояние до сети 201.36.14.0 стало равным 16, остается почти 30 секунд. Каждый маршрутизатор работает на основании своего внутреннего таймера, не синхронизируя работу по рассылке объявлений с другими маршрутизаторами. Поэтому весьма вероятно, что маршрутизатор R2 опередит маршрутизатор R1 и передаст ему свое сообщение раньше, чем R1 успеет передать новость о недостижимости сети 201.36.14.0. А в этом сообщении имеются данные, порожденные записью в таблице маршрутизации R2 (табл. 17.5).

Таблица 17.5. Таблица маршрутизации маршрутизатора R2

Номер сети	Адрес след. маршрутизатора	Порт	Расстояние
201.36.14.0	132.11.0.7	1	2

Эта запись, полученная от маршрутизатора R1, была корректна до отказа интерфейса 201.36.14.3, но теперь она устарела, причем маршрутизатор R2 об этом не знает.

Далее маршрутизатор R1 получает новую информацию о сети 201.36.14.0 — эта сеть достижима через маршрутизатор R2 с метрикой 2. Раньше маршрутизатор R1 также получал эту информацию от R2, но игнорировал ее, так как его собственная метрика для 201.36.14.0 была лучше. Теперь R1 должен принять данные о сети 201.36.14.0, полученные от R2, и заменить запись в таблице маршрутизации о недостижимости этой сети (табл. 17.6).

Таблица 17.6. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес след. маршрутизатора	Порт	Расстояние
201.36.14.0	132.11.0.101	2	3

В результате в сети образуется маршрутная петля: пакеты, направляемые узлам сети 201.36.14.0, станут передаваться маршрутизатором R2 маршрутизатору R1, а маршрутизатор R1 будет возвращать их маршрутизатору R2. IP-пакеты продолжат циркулировать по этой петле до тех пор, пока не истечет время жизни каждого пакета. Рассмотрим периоды времени, кратные времени жизни записей в таблицах маршрутизаторов.

- *Время 0–180 с.* После отказа интерфейса в маршрутизаторах R1 и R2 начинают сохраняться некорректные записи. Маршрутизатор R2 по-прежнему снабжает маршру-

тизатор R1 своей записью о сети 201.36.14.0 с метрикой 2, так как ее время жизни не истекло. Пакеты зацикливаются.

- *Время 180–360 с.* В начале этого периода у маршрутизатора R2 истекает время жизни записи о сети 201.36.14.0 с метрикой 2, так как маршрутизатор R1 в предыдущий период посылал ему сообщения о сети 201.36.14.0 с худшей метрикой, чем у R2, и они не могли подтвердить эту запись. Теперь маршрутизатор R2 принимает от маршрутизатора R1 запись о сети 201.36.14.0 с метрикой 3 и трансформирует ее в запись с метрикой 4. Маршрутизатор R1 не получает новых сообщений от маршрутизатора R2 о сети 201.36.14.0 с метрикой 2, поэтому время жизни его записи начинает уменьшаться. Пакеты продолжают зацикливаться.
- *Время 360–540 с.* У маршрутизатора R1 истекает время жизни записи о сети 201.36.14.0 с метрикой 3. Маршрутизаторы R1 и R2 опять меняются ролями: R2 снабжает R1 устаревшей информацией о пути к сети 201.36.14.0, уже с метрикой 4, которую R1 преобразует в метрику 5. Пакеты продолжают зацикливаться.

Если бы в протоколе RIP не было выбрано расстояние 16 в качестве недостижимого, то описанный процесс длился бы бесконечно (вернее, пока не была бы исчерпана разрядная сетка поля расстояния и при очередном наращивании расстояния было бы зафиксировано переполнение).

В результате маршрутизатор R2 на очередном этапе описанного процесса получает от маршрутизатора R1 метрику 15, которая после наращивания, превращаясь в метрику 16, фиксирует недостижимость сети. Таким образом, в нашем примере период нестабильной работы сети длится 36 минут!

Ограничение в 15 хопов сужает область применения протокола RIP до сетей, в которых число промежуточных маршрутизаторов не превышает 15. Для более масштабных сетей нужно применять другие протоколы маршрутизации, например OSPF, или разбивать сеть на автономные области.

Приведенный пример хорошо иллюстрирует главную причину нестабильности маршрутизаторов, работающих по протоколу RIP. Эта причина коренится в самом принципе работы дистанционно-векторных протоколов — использовании информации, полученной из «вторых рук». Действительно, маршрутизатор R2 передает маршрутизатору R1 информацию о достижимости сети 201.36.14.0, за достоверность которой он сам не отвечает.

ПРИМЕЧАНИЕ

Не следует думать, что при любых отказах интерфейсов и маршрутизаторов в сетях возникают маршрутные петли. Если бы маршрутизатор R1 успел передать сообщение о недостижимости сети 201.36.14.0 раньше ложной информации маршрутизатора R2, то маршрутная петля не образовалась бы. Так что маршрутные петли даже без дополнительных методов борьбы с ними возникают в среднем не более чем в половине потенциально возможных случаев.

Методы борьбы с ложными маршрутами в протоколе RIP

Хотя протокол RIP не в состоянии полностью исключить в сети переходные состояния, когда некоторые маршрутизаторы пользуются устаревшей информацией о несуществу-

ющих маршрутах, имеется несколько методов, позволяющих во многих случаях решать подобные проблемы.

Проблема с петлей, образующейся между соседними маршрутизаторами, надежно решается с помощью метода **расщепления горизонта**. Этот метод заключается в том, что маршрутная информация о некоторой сети, хранящаяся в таблице маршрутизации, никогда не передается тому маршрутизатору, от которого она получена.

Практически все сегодняшние маршрутизаторы, работающие по протоколу RIP, используют технику расщепления горизонта. Если бы маршрутизатор R2 в рассмотренном ранее примере поддерживал технику расщепления горизонта, то он бы не передал маршрутизатору R1 устаревшую информацию о сети 201.36.14.0, так как получил он ее именно от маршрутизатора R1.

Однако расщепление горизонта не помогает в тех случаях, когда петли образуются не двумя, а большим числом маршрутизаторов. Рассмотрим более детально ситуацию, которая возникнет в сети, приведенной на рис. 17.1, в случае потери связи маршрутизатора R1 с сетью 201.36.14.0. Пусть все маршрутизаторы этой сети поддерживают технику расщепления горизонта. В этой ситуации маршрутизаторы R2 и R3 не возвращают маршрутизатору данные о сети 201.36.14.0 с метрикой 2, так как они получили эту информацию от маршрутизатора R1. Однако они передают маршрутизатору информацию о достижимости сети 201.36.14.0 с метрикой 4 через себя, так как получили эту информацию по сложному маршруту, а не непосредственно от маршрутизатора R1. Например, маршрутизатор R2 получает эту информацию по цепочке R4-R3-R1, поэтому маршрутизатор R1 снова может быть обманут, пока каждый из маршрутизаторов в цепочке R3-R4-R2 не вычеркнет запись о достижимости сети 201.36.14.0.

Для предотвращения зацикливания пакетов по составным петлям при отказах связей применяются два других приема, называемые триггерными обновлениями и замораживанием изменений.

Прием **триггерных обновлений** состоит в том, что маршрутизатор, получив данные об изменении метрики до какой-либо сети, не ждет истечения периода передачи таблицы маршрутизации, а передает данные об изменившемся маршруте немедленно. Этот прием может во многих случаях предотвратить передачу устаревших сведений об отказавшем маршруте, но он перегружает сеть служебными сообщениями, поэтому триггерные объявления также делаются с некоторой задержкой. По этой причине возможна ситуация, когда регулярное обновление в каком-либо маршрутизаторе чуть опережает по времени приход триггерного обновления от предыдущего в цепочке маршрутизатора, и данный маршрутизатор успевает передать по сети устаревшую информацию о несуществующем маршруте.

Второй прием — **замораживание изменений** — позволяет исключить подобные ситуации. Он связан с введением тайм-аута на принятие новых данных о сети, которая только что стала недоступной. Этот тайм-аут предотвращает принятие устаревших сведений о некоем маршруте от тех маршрутизаторов, которые находятся на некотором расстоянии от отказавшей связи и передают устаревшие сведения о ее работоспособности. Предполагается, что в течение тайм-аута «замораживания изменений» эти маршрутизаторы вычеркнут данный маршрут из своих таблиц, так как не получат о нем новых записей и не будут распространять устаревшие сведения по сети.

Протокол OSPF

Протокол OSPF (Open Shortest Path First — выбор кратчайшего пути первым) является последним (он принят в 1991 году) протоколом, основанном на алгоритме состояния связей, и обладает многими особенностями, ориентированными на применение в больших гетерогенных сетях.

Два этапа построения таблицы маршрутизации

OSPF разбивает процедуру построения таблицы маршрутизации на два этапа, к первому относится построение и поддержание базы данных о состоянии связей сети, ко второму — нахождение оптимальных маршрутов и генерация таблицы маршрутизации.

Построение и поддержание базы данных о состоянии связей сети. Связи сети могут быть представлены в виде графа, в котором вершинами графа являются маршрутизаторы и подсети, а ребрами — связи между ними (рис. 17.2). Каждый маршрутизатор обменивается со своими соседями той информацией о графе сети, которой он располагают к данному моменту. Этот процесс похож на процесс распространения векторов расстояний до сетей в протоколе RIP, однако сама информация качественно иная — это информация о *топологии* сети. Сообщения, с помощью которых распространяется топологическая информация, называются **объявлениями о состоянии связей** (Link State Advertisement, LSA) сети. При транзитной передаче объявлений LSA маршрутизаторы не модифицируют информацию, как это происходит в дистанционно-векторных протоколах, в частности в RIP, а передают ее в неизменном виде. В результате все маршрутизаторы сети сохраняют в своей памяти идентичные сведения о текущей конфигурации графа связей сети.

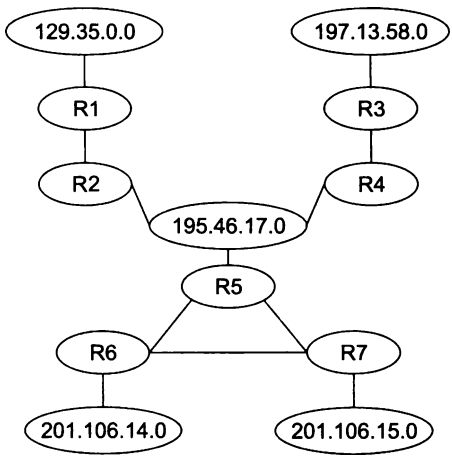


Рис. 17.2. Граф сети, построенный протоколом OSPF

Для контроля состояния связей и соседних маршрутизаторов маршрутизаторы OSPF передают друг другу особые сообщения HELLO каждые 10 секунд. Небольшой объем этих сообщений делает возможным частое тестирование состояния соседей и связей с ними.

В том случае, когда сообщения HELLO перестают поступать от какого-либо непосредственного соседа, маршрутизатор делает вывод о том, что состояние связи изменилось с работоспособного на неработоспособное, и вносит соответствующие коррективы в свою топологическую базу данных. Одновременно он отсылает всем непосредственным соседям объявление LSA об этом изменении, те также вносят исправления в свои базы данных и в свою очередь рассылают данное объявление LSA своим непосредственным соседям.

Нахождение оптимальных маршрутов и генерация таблицы маршрутизации. Задача нахождения оптимального пути на графе является достаточно сложной и трудоемкой. В протоколе OSPF для ее решения используется итеративный алгоритм Дейкстры. Каждый маршрутизатор сети, действуя в соответствии с этим алгоритмом, ищет оптимальные маршруты от своих интерфейсов до всех известных ему подсетей. В каждом найденном таким образом маршруте запоминается только один шаг — до следующего маршрутизатора. Данные об этом шаге и попадают в таблицу маршрутизации.

Если состояние связей в сети изменилось и произошла корректировка графа сети, каждый маршрутизатор заново ищет оптимальные маршруты и корректирует свою таблицу маршрутизации. Аналогичный процесс происходит и в том случае, когда в сети появляется новая связь или новый сосед, объявляющие о себе с помощью своих сообщений HELLO. При работе протокола OSPF конвергенция таблиц маршрутизации к новому согласованному состоянию происходит достаточно быстро, быстрее, чем в сетях, в которых работают дистанционно-векторные протоколы. Это время состоит из времени распространения по сети объявления LSA и времени работы алгоритма Дейкстры, который обладает быстрой сходимостью. Однако вычислительная сложность этого алгоритма предъявляет высокие требования к мощности процессоров маршрутизаторов.

Когда состояние сети не меняется, то объявления о связях не генерируются, топологические базы данных и таблицы маршрутизации не корректируются, что экономит пропускную способность сети и вычислительные ресурсы маршрутизаторов. Однако у этого правила есть исключение: каждые 30 минут маршрутизаторы OSPF обмениваются всеми записями базы данных топологической информации, то есть синхронизируют их для более надежной работы сети. Так как этот период достаточно большой, то данное исключение незначительно сказывается на загрузке сети.

Метрики

При поиске оптимальных маршрутов протокол OSPF по умолчанию использует метрику, учитывающую пропускную способность каналов связи. Кроме того, допускается применение двух других метрик, учитывающих задержки и надежность передачи пакетов каналами связи. Для каждой из метрик протокол OSPF строит *отдельную* таблицу маршрутизации. Выбор нужной таблицы происходит в зависимости от значений битов TOS в заголовке пришедшего IP-пакета. Если в пакете бит D (Delay — задержка) установлен в 1, то для этого пакета маршрут должен выбираться из таблицы, в которой содержатся маршруты, имеющие минимальную задержку. Аналогично, пакет с установленным битом T (Throughput — пропускная способность) должен маршрутизироваться по таблице, построенной с учетом пропускной способности каналов, а установленный в единицу бит R (Reliability — надежность) указывает на то, что должна использоваться таблица, для построения которой критерием оптимизации служит надежность доставки.

Протокол OSPF поддерживает стандартные для многих протоколов (например, для протокола покрывающего дерева) значения расстояний для метрики, отражающей пропускную способность: так, для сети Ethernet она равна 10, для Fast Ethernet — 1, для канала T-1[1], обладающего пропускной способностью 1,544 Мбит/с, — 65, для канала с пропускной способностью 56 Кбит/с — 1785. При наличии высокоскоростных каналов, таких как Gigabit Ethernet или STM-16/64, администратору нужно задать другую шкалу скоростей, назначив единичное расстояние наиболее скоростному каналу.

При выборе оптимального пути на графе с каждым ребром графа связывается метрика, которая добавляется к пути, если данное ребро в него входит. Пусть в приведенном на рис. 17.2 примере маршрутизатор R5 связан с маршрутизаторами R6 и R7 каналами T-1, а маршрутизаторы R6 и R7 связаны между собой каналом 56 Кбит/с. Тогда R7 определит оптимальный маршрут до сети 201.106.14.0 как составной, проходящий сначала через R5, а затем через R6, поскольку у этого маршрута метрика равна 65 + 65 = 130 единиц. Непосредственный маршрут через R6 не будет оптимальным, так как его метрика равна 1785.

Протокол OSPF разрешает хранить в таблице маршрутизации несколько маршрутов к одной сети, если они обладают равными метриками. В таких случаях маршрутизатор может работать в режиме баланса загрузки маршрутов, отправляя пакеты попеременно по каждому из маршрутов.

К сожалению, вычислительная сложность протокола OSPF быстро растет с увеличением размера сети. Для преодоления этого недостатка в протоколе OSPF вводится понятие **области сети**. Маршрутизаторы, принадлежащие некоторой области, строят граф связей только для этой области, что упрощает задачу. Между областями информация о связях не передается, а пограничные для областей маршрутизаторы обмениваются только информацией об адресах сетей, имеющихся в каждой из областей, и *расстоянием от пограничного маршрутизатора до каждой сети*. При передаче пакетов между областями выбирается один из пограничных маршрутизаторов области, а именно тот, у которого расстояние до нужной сети меньше.

Протокол **IS-IS** (Intermediate System to Intermediate System) функционально близок к протоколу OSPF. Существуют различные версии IS-IS, рассчитанные на работу в различных стеках протоколов и способные переносить в своих сообщениях адресную информацию различного типа. Наиболее популярной является версия IS-IS для стека TCP/IP, недавно появилась версия IS-IS для работы в сетях Ethernet (см. главу 21).

Маршрутизация в неоднородных сетях

Взаимодействие протоколов маршрутизации

В одной и той же сети могут одновременно работать несколько разных протоколов маршрутизации (рис. 17.3). Это означает, что на некоторых (не обязательно всех) маршрутизаторах сети установлено и функционирует несколько протоколов маршрутизации, но при этом, естественно, через сеть взаимодействуют только одноименные протоколы. То есть если маршрутизатор 1 поддерживает, например, протоколы RIP и OSPF, маршрутизатор 2 — только RIP, а маршрутизатор 3 — только OSPF, то маршрутизатор 1 будет взаимодейство-

[1] T-1 — это цифровой канал технологии PDH, рассматривавшейся в главе 10.

вать с маршрутизатором 2 по протоколу RIP, с маршрутизатором 2 — по OSPF, а маршрутизаторы 2 и 3 вообще непосредственно друг с другом взаимодействовать не смогут.

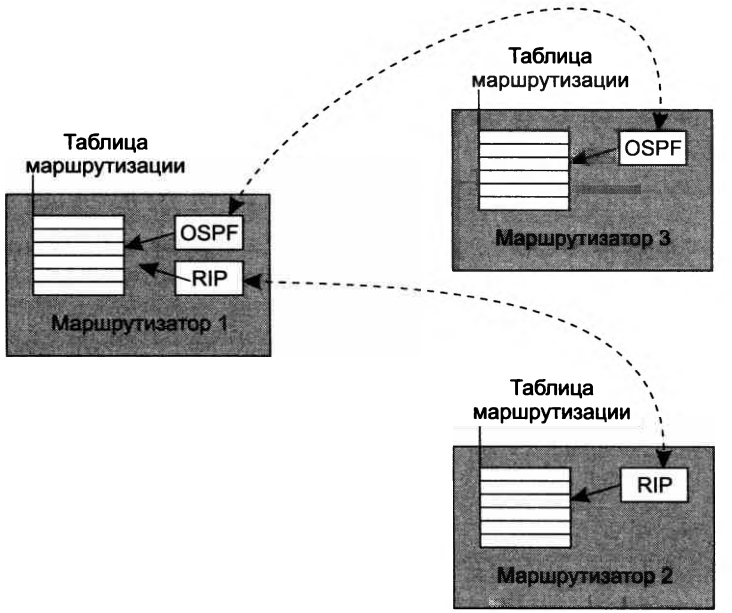


Рис. 17.3. Применение нескольких протоколов маршрутизации в одной сети

В маршрутизаторе, который поддерживает одновременно несколько протоколов, каждая запись в таблице является результатом работы одного из этих протоколов. Если информация о некоторой сети появляется от нескольких протоколов, то для однозначности выбора маршрута (а данные разных протоколов могут вести к разным рациональным маршрутам) устанавливаются *приоритеты протоколов маршрутизации*. Обычно предпочтение отдается протоколам LSA как располагающим более полной информацией о сети по сравнению с протоколами DVA. В некоторых ОС в формах вывода на экран и печать в каждой записи таблицы маршрутизации имеется отметка о том, с помощью какого протокола маршрутизации эта запись получена. Но даже если эта отметка на экран и не выводится, она обязательно имеется во внутреннем представлении таблицы маршрутизации.

По умолчанию каждый протокол маршрутизации, работающий на определенном маршрутизаторе, распространяет только «собственную» информацию, то есть ту информацию, которая была получена данным маршрутизатором по данному протоколу. Например, если о маршруте к некоторой сети маршрутизатор узнал по протоколу RIP, то и распространять по сети объявления об этом маршруте он будет с помощью протокола RIP.

Однако такой «избирательный» режим работы маршрутизаторов ставит невидимые барьеры на пути распространения маршрутной информации, создавая в составной сети области взаимной недостижимости. Задача маршрутизации решалась бы эффективнее, если бы маршрутизаторы могли обмениваться маршрутной информацией, полученной разными протоколами маршрутизации. Такая возможность реализуется в особом режиме работы маршрутизатора, называемом **режимом перераспределения маршрутов**. Этот

режим позволяет одному протоколу маршрутизации использовать не только «свои», но и «чужие» записи таблицы маршрутизации, полученные с помощью другого протокола маршрутизации, указанного при конфигурировании.

Как видим, применение нескольких протоколов маршрутизации даже в пределах небольшой составной сети — дело непростое, от администратора требуется провести определенную работу по конфигурированию каждого маршрутизатора. Очевидно, что для крупных составных сетей нужно качественно иное решение.

Внутренние и внешние шлюзовые протоколы

Такое решение было найдено для самой крупной на сегодня составной сети — Интернета. Это решение базируется на понятии автономной системы.

Автономная система (Autonomous System, AS — это совокупность сетей под единым административным управлением, обеспечивающим общую для всех входящих в автономную систему маршрутизаторов политику маршрутизации.

Обычно автономной системой управляет один поставщик услуг Интернета, самостоятельно выбирая, какие протоколы маршрутизации должны использоваться в некоторой автономной системе и каким образом между ними должно выполняться перераспределение маршрутной информации. Крупные поставщики услуг и корпорации могут представить свою составную сеть как набор нескольких автономных систем. Регистрация автономных систем происходит централизованно, как и регистрация IP-адресов и DNS-имен. Номер автономной системы состоит из 16 разрядов и никак не связан с префиксами IP-адресов входящих в нее сетей.

В соответствии с этой концепцией Интернет выглядит как набор взаимосвязанных автономных систем, каждая из которых состоит из взаимосвязанных сетей (рис. 17.4), соединенных **внешними шлюзами**.

Основная цель деления Интернета на автономные системы — обеспечение многоуровневого подхода к маршрутизации. До введения автономных систем предполагался двухуровневый подход, то есть сначала маршрут определялся как *последовательность сетей*, а затем вел непосредственно к заданному узлу в конечной сети (именно этот подход мы использовали до сих пор).

С появлением автономных систем появляется третий, верхний, уровень маршрутизации — теперь сначала маршрут определяется как *последовательность автономных систем*, затем — как *последовательность сетей* и только потом ведет к конечному узлу.

Выбор маршрута между автономными системами осуществляют внешние шлюзы, использующие особый тип протокола маршрутизации, так называемый **внешний шлюзовой протокол** (Exterior Gateway Protocol, EGP). В настоящее время для работы в такой роли сообщество Интернета утвердило стандартный **пограничный шлюзовой протокол** версии 4 (Border Gateway Protocol, BGPv4). В качестве адреса следующего маршрутизатора в протоколе BGPv4 указывается адрес точки входа в соседнюю автономную систему.

За *маршрут внутри автономной системы* отвечают **внутренние шлюзовые протоколы** (Interior Gateway Protocol, IGP). К числу IGP относятся знакомые нам протоколы RIP,

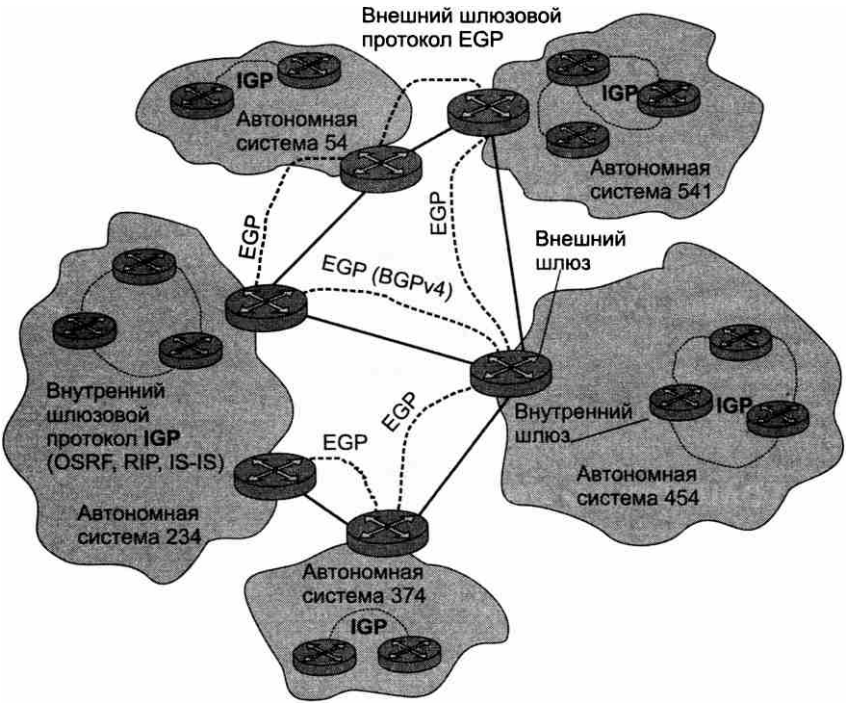


Рис. 17.4. Автономные системы Интернета

OSPF и IS-IS. В случае транзитной автономной системы эти протоколы указывают точную последовательность маршрутизаторов от точки входа в автономную систему до точки выхода из нее.

ПРИМЕЧАНИЕ

Внутри каждой автономной системы может применяться любой из существующих протоколов маршрутизации, в то время как между автономными системами всегда применяется один и тот же протокол, являющийся своеобразным языком «эсперанто», на котором автономные системы общаются между собой.

Концепция автономных систем скрывает от администраторов магистралей Интернета проблемы маршрутизации пакетов на более низком уровне — уровне сетей. Для администратора магистрали не важно, какие протоколы маршрутизации применяются внутри автономных систем, для него существует единственный протокол маршрутизации — BGP.

Протокол BGP

Пограничный (внешний) шлюзовой протокол (Border Gateway Protocol, BGP) в версии 4 является сегодня основным протоколом обмена маршрутной информацией между автономными системами Интернета.

BGP успешно работает при любой топологии связей между автономными системами, что соответствует современному состоянию Интернета.

Поясним основные принципы работы BGP на примере (рис. 17.5).

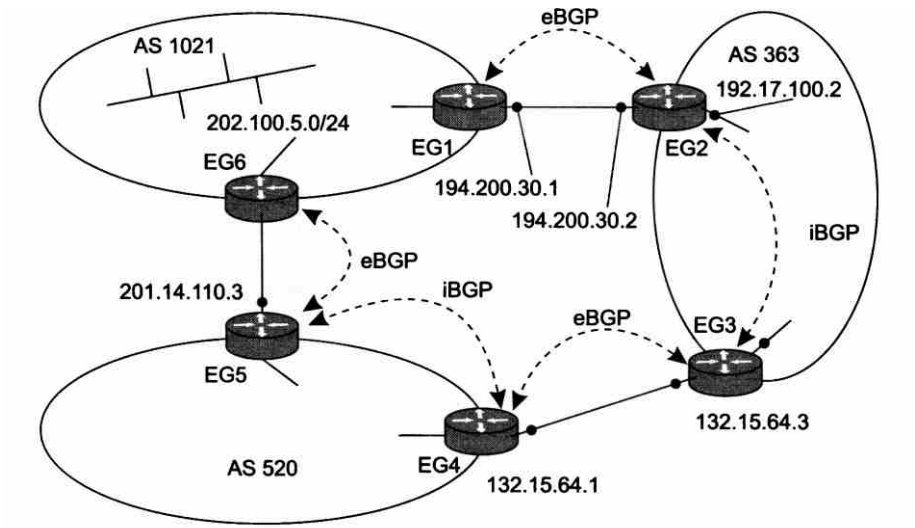


Рис. 17.5. Поиск маршрута между автономными системами с помощью протокола BGP

В каждой из трех автономных систем (AS 1021, AS 363 и AS 520) имеется несколько маршрутизаторов, исполняющих роль внешних шлюзов. На каждом из них работает протокол BGP, с помощью которого они общаются между собой.

Маршрутизатор взаимодействует с другими маршрутизаторами по протоколу BGP только в том случае, если администратор *явно* указывает при конфигурировании, что эти маршрутизаторы являются его *соседями*. Например, маршрутизатор EG1 в рассматриваемом примере будет взаимодействовать по протоколу BGP с маршрутизатором EG2 не потому, что эти маршрутизаторы соединены двухточечным каналом, а потому, что при конфигурировании маршрутизатора EG1 в качестве соседа ему был указан маршрутизатор EG2 (с адресом 194.200.30.2). Аналогично, при конфигурировании маршрутизатора EG2 его соседом был назначен маршрутизатор EG1 (с адресом 194.200.30.1).

Такой способ взаимодействия удобен в ситуации, когда маршрутизаторы, обменивающиеся маршрутной информацией, принадлежат разным поставщикам услуг (ISP). Администратор ISP может решать, с какими автономными системами он будет обмениваться трафиком, а с какими нет, задавая список соседей для своих внешних шлюзов. Протоколы RIP и OSPF, разработанные для применения внутри автономной системы, обмениваются маршрутной информацией со всеми маршрутизаторами, находящимися в пределах их непосредственной досягаемости (по локальной сети или через двухточечный канал). Это означает, что информация обо всех сетях появляется в таблице маршрутизации каждого маршрутизатора, так что каждая сеть оказывается достижимой для каждой. В корпоративной сети это

нормальная ситуация, а в сети ISP нет, поэтому протокол BGP и исполняет здесь особую роль, поддерживая разнообразные и гибкие политики маршрутизации, говорящие о том, каким соседям передавать маршрутные объявления и о каких сетях им в этих объявлениях сообщать, а также от каких соседей и о каких сетях можно принимать маршрутные объявления. Политика маршрутизации провайдера отражает условия по взаимной передаче трафика, имеющиеся в *соглашениях об уровне обслуживания* (SLA), или в *пиринговых соглашениях* (от *peering* — отношения равных субъектов), которые провайдер заключает с другими провайдерами.

Для установления сеанса с указанными соседями маршрутизаторы BGP используют протокол TCP (порт 179). При установлении BGP-сеанса могут применяться разнообразные способы аутентификации маршрутизаторов, повышающие безопасность работы автономных систем.

Основным сообщением протокола BGP является сообщение UPDATE (обновить), с помощью которого маршрутизатор сообщает маршрутизатору соседней автономной системы о достижимости сетей, относящихся к его собственной автономной системе. Само название этого сообщения говорит о том, что это — триггерное объявление, которое посылается соседу только тогда, когда в автономной системе что-нибудь резко меняется: появляются новые сети или новые пути к сетям либо же, напротив, исчезают существовавшие сети или пути.

В одном сообщении UPDATE можно объявить об одном новом маршруте или аннулировать несколько маршрутов, переставших существовать. Под маршрутом в BGP понимается последовательность автономных систем, которую нужно пройти на пути к указанной в адресе сети. Более формально информация о маршруте (BGP Route) к сети (Network/Mask_length) выглядит так:

BGP Route = AS_Path; NextHop; Network/Mask_length;

Здесь AS_Path — набор номеров автономных систем, NextHop — IP-адрес маршрутизатора, через который нужно передавать пакеты в сеть Network/Mask_length. Например, если маршрутизатор EG1 хочет объявить маршрутизатору EG2 о том, что в AS 1021 появилась новая сеть 202.100.5.0/24, то он формирует такое сообщение:

AS 1021; 194.200.30.1; 202.100.5.0/24,

Затем он передает это сообщение маршрутизатору EG2 автономной системы AS 363 (с которым у него, конечно, должен быть установлен BGP-сеанс).

Маршрутизатор EG2, получив сообщение UPDATE, запоминает в своей таблице маршрутизации информацию о сети 202.100.5.0/24 вместе с адресом следующего маршрутизатора 194.200.30.1 и отметкой о том, что эта информация была получена по протоколу BGP. Маршрутизатор EG2 обменивается маршрутной информацией с внутренними шлюзами системы AS 363 по какому-либо протоколу группы IGP, например OSPF. Если у EG2 установлен *режим перераспределения маршрутов* BGP в маршруты OSPF, то все внутренние шлюзы AS 363 узнают о существовании сети 202.100.5.0/24 с помощью объявления OSPF, которое будет внешним. В качестве адреса следующего маршрутизатора маршрутизатор EG2 начнет теперь объявлять адрес собственного внутреннего интерфейса, например 192.17.100.2.

Однако для распространения сообщения о сети 202.100.5.0/24 в другие автономные системы, например в AS 520, протокол OSPF использоваться не может. Маршрутизатор EG3,

связанный с маршрутизатором EG4 автономной системы 520, должен применять протокол BGP, генерируя сообщение UPDATE нужного формата. Для решения этой задачи он не может задействовать информацию о сети 202.100.5.0/24, полученную от протокола OSPF через один из своих внутренних интерфейсов, так как она имеет другой формат и не содержит, например, сведений о номере автономной системы, в которой находится эта сеть.

Проблема решается за счет того, что маршрутизаторы EG2 и EG3 также устанавливают между собой BGP-сеанс, хотя они и принадлежат одной и той же автономной системе. Такая реализация протокола BGP называется **внутренней версией BGP** (Interior BGP, **iBGP**), в отличие от основной, **внешней версии** (Exterior BGP, **eBGP**). В результате маршрутизатор EG3 получает нужную информацию от маршрутизатора EG2 и передает ее внешнему соседу — маршрутизатору EG4. При формировании нового сообщения UPDATE маршрутизатор EG3 трансформирует сообщение, полученное от маршрутизатора EG2, добавляя в список автономных систем собственную автономную систему AS 520, а полученный адрес следующего маршрутизатора заменяет адресом собственного интерфейса:

AS 363, AS 1021; 132.15.64.3; 202.100.5.0/24.

Номера автономных систем позволяют исключить зацикливание сообщений UPDATE. Например, когда маршрутизатор EG5 передаст сообщение о сети 202.100.5.0/24 маршрутизатору EG6, то последний не станет его использовать, так как оно будет иметь вид:

AS 520, AS 363, AS 1021; 201.14.110.3; 202.100.5.0/24.

Так как в списке автономных систем уже есть номер собственной автономной системы, очевидно, что сообщение зациклилось.

Протокол BGP используется сегодня для обмена маршрутной информацией не только между автономными системами, но и внутри них.

Групповое вещание

Групповое вещание, применяемое ранее только в радио- и телевизионных сетях, в последние годы все шире внедряется в компьютерные сети. Наиболее востребована эта технология в Интернете, который представляет собой идеальную среду для массового распространения по подписке мультимедийной информации — аудиозаписей, видеофильмов, информационных дайджестов и т. п. Приложения, реализующие такого рода услуги, требуют наличия механизма доставки одной и той же информации определенному кругу абонентов сети.

Концепция **группового вещания** (multicast) нашла свое воплощение в ряде спецификаций протоколов группового взаимодействия в Интернете. В 1992 году появилась экспериментальная магистраль MBone, которая объединила 20 сетей через Интернет. С помощью этой магистрали была проведена первая аудиоконференция в Сан-Диего, голосовой поток которой был адресован *группе*, образованной из членов IETF по всему миру.

Стандартная модель группового вещания IP

Основной целью группового вещания является создание эффективного механизма передачи данных от одного источника нескольким получателям. Для решения этой задачи

могут использоваться несколько подходов, например индивидуальная рассылка, широковещательная рассылка, привлечение сервисов прикладного уровня.

При *индивидуальной рассылке* (unicast) на основе уникальных адресов источник данных, которые надо доставить некоторой группе узлов, генерирует их в количестве экземпляров, равном количеству узлов-получателей, состоящих в данной группе (рис. 17.6). То есть передача по принципу «один ко многим» сводится к нескольким передачам «один к одному». Очевидно, что передача нескольких идентичных копий на участках, где маршруты к разным членам группы перекрываются (это особенно характерно для начальных участков), приводит к избыточному трафику.

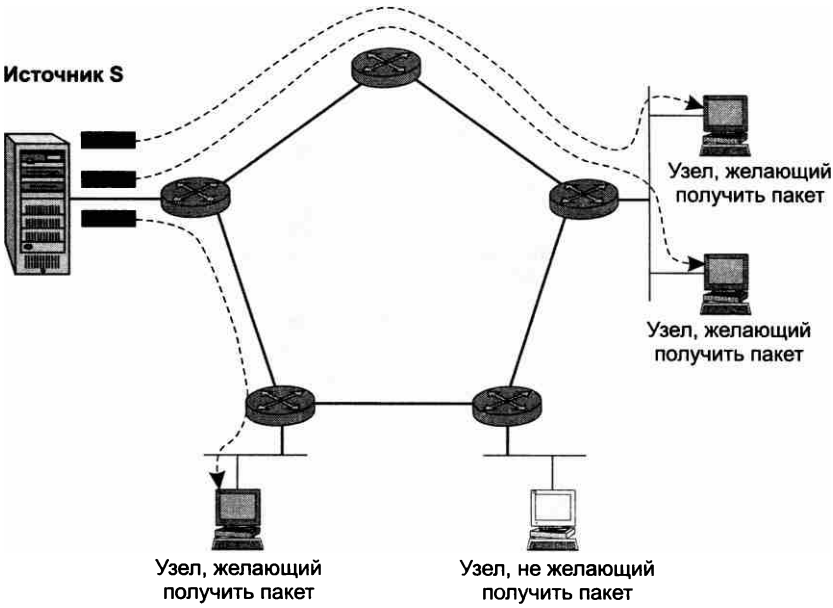


Рис. 17.6. Групповая доставка на основе индивидуальных адресов

При *широковещательной рассылке* (broadcast) станция направляет пакеты, используя широковещательные адреса (рис. 17.7). В этой схеме для того, чтобы доставить данные группе узлов-получателей, источник генерирует один экземпляр данных, но снабжает этот экземпляр широковещательным адресом, который диктует маршрутизаторам сети копировать данные и рассылать их всем конечным узлам независимо от того, «заинтересованы» узлы в получении этих данных или нет. В этом случае, как и в предыдущем, существенная доля трафика является избыточной.

В случае *привлечения сервисов прикладного уровня* функции по обеспечению групповой доставки перекладываются на самих членов группы. То есть, как показано на рис. 17.8, источник генерирует один экземпляр данных и, используя индивидуальный адрес, передает данные одному из членов группы, который генерирует копию и направляет ее другому члену группы, и т. д. Перевод решения задачи с нижних транспортных уровней на прикладной уровень повышает суммарные накладные расходы сети на реализацию групповой доставки и делает этот механизм менее гибким.

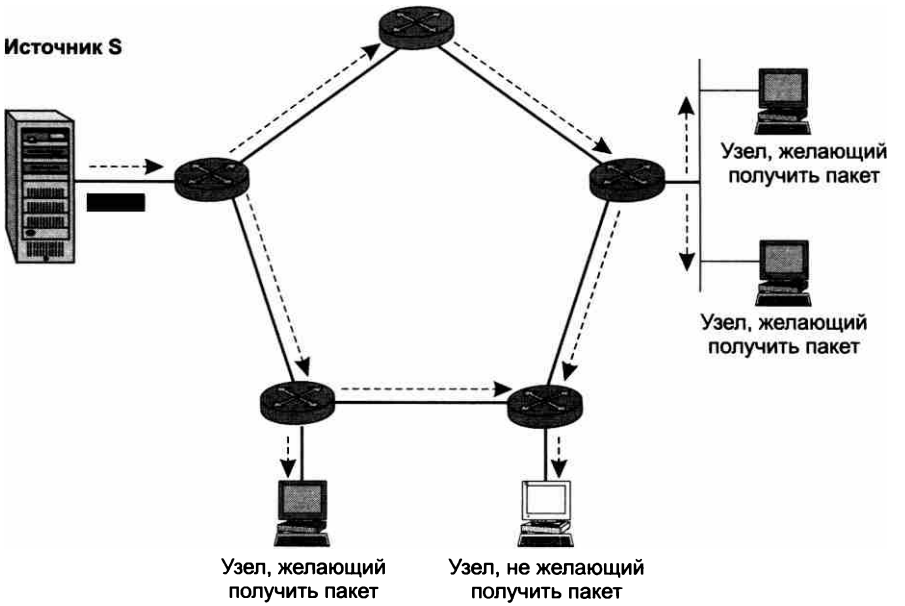


Рис. 17.7. Групповая доставка на основе широковещательного адреса

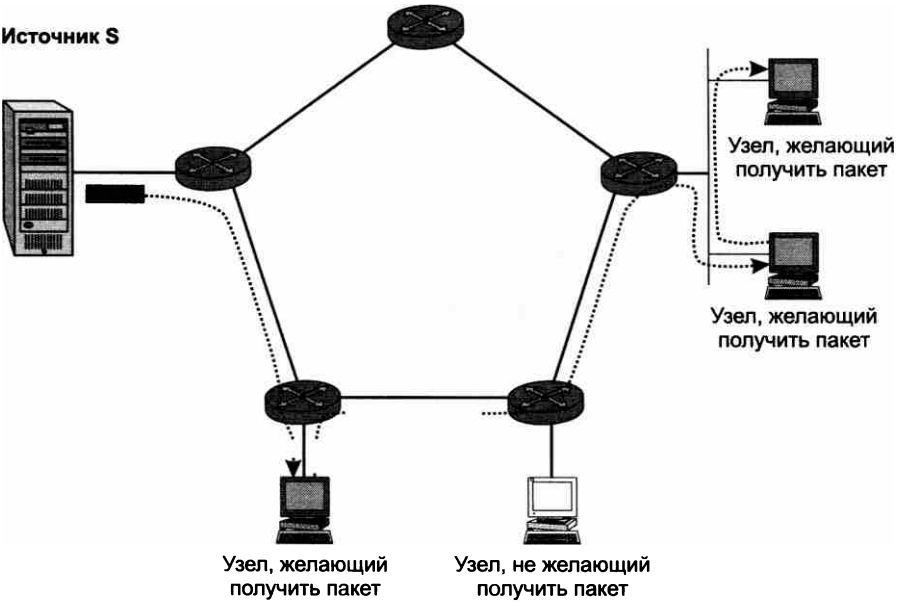


Рис. 17.8. Групповая доставка на основе сервисов прикладного уровня

Таким образом, традиционные механизмы доставки пакетов стека TCP/IP мало пригодны для поддержки группового вещания. В такой ситуации наиболее эффективным решением является использование специально разработанного механизма группового вещания, ориентированного на сокращение избыточного трафика и накладных расходов сети.

Главная идея группового вещания состоит в следующем: источник генерирует только один экземпляр сообщения с групповым адресом, которое затем, по мере перемещения по сети, копируется на каждой из «развилок», ведущих к тому или иному члену группы, указанной в адресе данного сообщения (рис. 17.9).

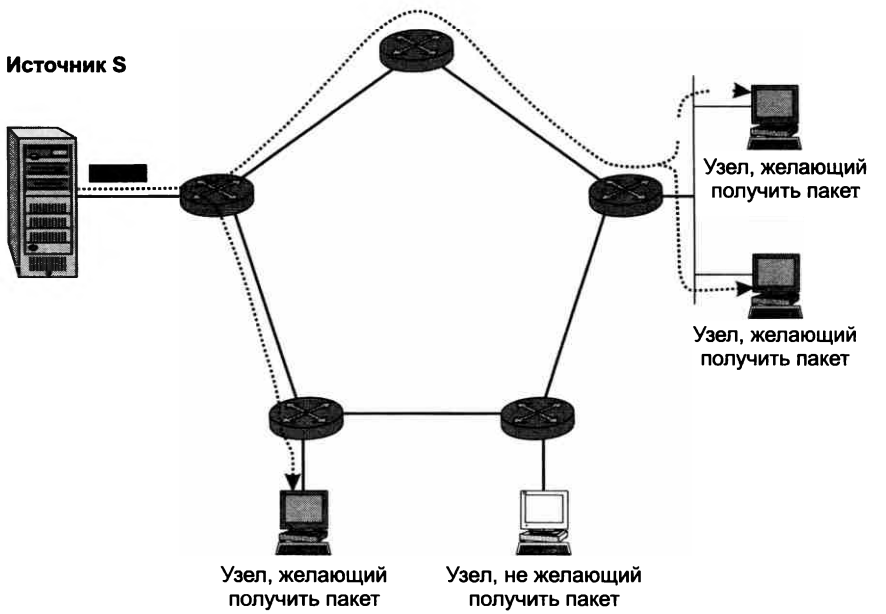


Рис. 17.9. Схема группового вещания

В конце концов пакет с групповым адресом достигает маршрутизатора, к которому непосредственно подключена сеть с хостами — членами данной группы. Напомним, что у хостов, относящихся к той или иной группе, интерфейс наряду с индивидуальным адресом имеет еще и групповой адрес — адрес класса D, называемый также адресом группового вещания. Интерфейс может иметь даже несколько групповых адресов — по числу групп, в которых состоит данный хост.

Как и в случае обычной маршрутизации на базе индивидуальных адресов, маршрутизатор упаковывает пакет с групповым адресом в кадр канального уровня (той технологии, которая используется в данной локальной сети, например Ethernet), снабжая его групповым MAC-адресом, соответствующим групповому IP-адресу пакета[1]. Кадр с пакетом группового вещания поступает в локальную сеть, распознается и захватывается интерфейсами хостов, являющихся членами данной группы.

---

[1] Об отображении групповых IP-адресов на групповые MAC-адреса см. далее в разделе «Протокол IGMP».

При таком подходе данные рассылаются только тем узлам, которые заинтересованы в их получении. Функция репликации группового сообщения и продвижения копий в сторону членов группы возлагается на маршрутизаторы, для чего *они должны быть оснащены соответствующими программно-аппаратными средствами*. Такой режим экономит пропускную способность за счет передачи только того трафика, который необходим.

Стив Диринг (Steve Deering) — один из главных идеологов группового вещания — сформулировал несколько принципиальных положений, регламентирующих поведение конечных узлов сети, которые являются источниками и получателями группового трафика.

- *Дейтаграммный подход*. Источник может посылать пакеты UDP/IP в любое время без необходимости регистрировать или планировать передачи, реализуя сервис «по возможности».
- *Открытые группы*. Источники должны знать только групповой адрес. Они не должны знать членов группы и не обязательно должны быть членами той группы, которой они посылают данные. Группа может быть образована узлами, принадлежащими к разным IP-сетям и подсетям. Группа может иметь любое число источников данных.
- *Динамические группы*. Хосты могут присоединяться к группам или покидать группы без необходимости регистрации, синхронизации или переговоров с каким-либо централизованным элементом группового управления. Членство в группе является динамическим, поскольку хосты могут присоединиться к группе или выйти из группы в любой момент времени, к тому же они могут быть членами нескольких групп.

В этих концептуальных положениях Диринг говорит о правилах для конечных узлов, выполняющих функции источников и получателей, но не обсуждает требования к маршрутизации группового трафика. Он также не определяет механизмы обеспечения качества обслуживания, безопасности или назначения адресов.

Адреса группового вещания

Ранее в главе 14, изучая типы IP-адресов, мы отмечали, что адреса IPv4 из диапазона 224.0.0.0–239.255.255.255 относятся к классу D и зарезервированы для группового вещания.

Адреса из этого диапазона используются:

- для идентификации групп;
- для идентификации адресов источников группового вещания (в рамках модели SSM);
- для административных нужд при реализации группового вещания.

В общем случае адреса используются динамически, то есть если после остановки вещания источник снова начинает передачу, то он в общем случае может задействовать новый адрес группового вещания. Так называемые *хорошо известные* источники обычно наделяются постоянным групповым адресом.

Информацию о том, какие адреса уже закреплены для исполнения некоторой постоянной роли, а также о том, как использовать адресное пространство адресов класса D, дает документ RFC 3171 полномочной организации по цифровым адресам Интернета (Internet Assigned Numbers Authority, IANA).

(S) *Структурирование адресного пространства группового вещания*

На основе описанной концепции для стека TCP/IP был разработан ряд протоколов, с помощью которых можно организовать групповое вещание с различной степенью эффективности. Эти протоколы делятся на две категории. В первую входит один протокол — **протокол IGMP**, с помощью которого, во-первых, хосты сообщают о своем «желании»[1] присоединиться к некоторой группе, во-вторых, маршрутизатор узнает о принадлежности хостов в непосредственно подключенных к нему подсетях к той или иной группе. Вторую группу составляют **протоколы маршрутизации группового вещания**, которые необходимы для продвижения через сеть произвольной конфигурации пакетов, несущих в себе информацию для групповых получателей.

Протокол IGMP

Протокол группового управления в Интернете (Internet Group Management Protocol, IGMP), разработанный в 1989 году, используется исключительно при взаимодействии непосредственно связанных друг с другом маршрутизатора и хоста, когда последний выступает (или желает выступать) в роли получателя трафика группового вещания.

ПРИМЕЧАНИЕ

Источник не нуждается в протоколе IGMP. Любой компьютер, подключенный к Интернету, может стать источником группового вещания, при этом ему не требуется никакое дополнительное программное обеспечение, кроме того, которое включено в состав обычной реализации стека TCP/IP.

К основным функциям протокола IGMP относятся оповещение маршрутизатора о желании хоста быть включенным в группу и опрос членов группы.

Оповещение маршрутизатора о желании хоста быть включенным в группу. Чтобы стать получателем групповых данных, узел должен «выразить» свою заинтересованность маршрутизатору, к которому непосредственно подсоединена его сеть. Для этого хост должен установить взаимодействие с маршрутизатором по протоколу IGMP. Версия IGMP для хоста непосредственно зависит от типа операционной системы, установленной на хосте. Так, ранние версии Windows (Windows 95) поддерживали только версию IGMPv1, более поздние (Windows 2000) — версию IGMPv2, а начиная с Windows XP поддерживается версия IGMPv3. Протоколы IGMPv2 и IGMPv3 поддерживаются во многих версиях Mac OS, Linux, UNIX-подобных операционных системах.

Опрос членов группы. Для выполнения этой функции один из маршрутизаторов локальной сети выбирается доминирующим. Доминирующий маршрутизатор средствами протокола IGMP периодически опрашивает все системы (групповой адрес 224.0.0.1) в непосредственно присоединенных к нему подсетях, проверяя, активны ли члены всех известных ему групп. Остальные (невыбранные) маршрутизаторы прослушивают сеть, и если обнаруживают отсутствие сообщений-запросов в течение некоторого периода (обычно 250 секунд), то повторяют процедуру выбора нового доминирующего маршрутизатора.

В IGMPv2 определено три типа сообщений:

- *Запрос о членстве* (membership query). С помощью этого сообщения маршрутизатор пытается узнать, в каких группах состоят хосты в локальной сети, присоединенной к ка-

[1] Точнее, о «желании» приложения, выполняющегося на этом хосте, получать трафик, направляемый той или иной группе.

кому-либо его интерфейсу. Запрос о членстве существует в двух вариантах: в одном из них маршрутизатор делает общий запрос обо всех группах, в другом его интересует информация только о какой-то конкретной группе, адрес которой указывается в запросе.

- *Отчет о членстве* (membership report). Этим сообщением хосты отвечают маршрутизатору, который послал в сеть запрос о членстве. В сообщении содержится информация об адресе группы, в которой они состоят. Маршрутизатор, являясь членом всех групп, получает сообщения, направленные на любой групповой адрес. Для маршрутизатора, получающего ответные сообщения, важен только факт наличия членов той или иной группы (групп), а не принадлежность конкретных хостов конкретным группам. Этот факт будет использован другими маршрутизаторами сети для продвижения пакетов группового вещания в ту часть сети, за которую «отвечает» данный маршрутизатор. Отчет о членстве хост может послать не только в ответ на запрос маршрутизатора, но и *по собственной инициативе*, когда он пытается присоединиться к определенной группе. После такого сообщения хост может рассчитывать на то, что трафик для этой группы действительно будет доставляться в сеть, к которой этот хост принадлежит.
- *Покинуть группу* (leave group). Это сообщение хост *может* использовать, чтобы сигнализировать «своему» маршрутизатору о желании покинуть определенную группу, в которой он до этого состоял. Получив это сообщение, маршрутизатор посылает специфический запрос о членстве членам только этой конкретной группы, и если не получает на него ответа (что говорит от том, что это последний хост в группе), то перестает передавать трафик группового вещания для этой группы. Слово «может» означает в данном случае, что хост может быть исключен из группы, просто не отвечая маршрутизатору на запрос о членстве (такой подход реализован в протоколе IGMPv1). Тогда маршрутизатор будет продолжать передавать нежелательный трафик группового вещания до тех пор, пока не истечет некоторый период времени с момента поступления последнего отчета о членстве. Такой подход может значительно удлинить период скрытого нахождения хоста в состоянии выхода из группы, что снижает эффективность работы сети.

Сообщения с запросами о членстве посылаются маршрутизатором регулярно с некоторой частотой. На каждом из интерфейсов с установленными средствами IGMP маршрутизаторами поддерживаются кэш-таблицы групп. Кэш-таблица содержит список всех групп, в составе которых есть хотя бы один член. Для каждой строки таблицы установлен тайм-аут. Маршрутизатор регулярно посылает запросы (по умолчанию — каждые 125 секунд), чтобы проверить, что в каждой группе еще имеются члены. Если для некоторой группы ответ не поступает в течение установленного для нее тайм-аута, то соответствующая строка удаляется из кэш-таблицы и маршрутизатор считает, что членов этой группы в сети больше нет.

Локальная сеть может иметь несколько хостов, заинтересованных в получении трафика одной и той же группы, но маршрутизатору достаточно подтверждения только от одного хоста, чтобы продолжать передачу трафика в сеть для этой группы. При использовании протокола IGMPv1 или IGMPv2 для ограничения числа ответов хостов на запрос маршрутизатора любой хост, состоящий в группе, вместо того чтобы немедленно ответить на запрос, сначала ждет в течение некоторого интервала времени, не появится ли в сети ответ какого-нибудь другого хоста. Если по истечении этого времени он так и не смог дождаться появления в сети ответа другого хоста, то он посылает маршрутизатору собственный отчет о членстве. (Если же используется протокол IGMPv3, то никаких пауз не устанавливается и хосты сразу генерируют сообщения о членстве.)

Основываясь на информации, полученной с помощью IGMP, маршрутизаторы могут определять, в какие подключенные к ним сети необходимо передавать групповой трафик.

Все типы IGMP-сообщений имеют длину 8 байт и состоят из четырех полей. В зависимости от версии протокола IGMP назначение полей может несколько меняться. На рис. 17.10 показана структура сообщения для версии IGMPv2.

Рис. 17.10. Структура IGMP-сообщения

Поле максимального времени ответа используется хостами для вычисления времени задержки ответа. Время задержки выбирается случайным образом из интервала от нуля до значения, заданного в этом поле.

Заметим, что поле адреса группового вещания в IGMP-сообщении *не содержит* адреса назначения, оно несет в себе информацию, по-разному используемую в разных типах сообщений. Например, маршрутизатор, посылая запрос о членстве, помещает в этом поле нули, а хост в сообщениях «Отчет о членстве» и «Покинуть группу» помещает в это поле адрес группы, в которую он хочет вступить или которую он хочет покинуть соответственно.

ПРИМЕЧАНИЕ

Чтобы хост смог получать трафик группового вещания, недостаточно установить на нем протокол IGMP, с помощью которого хост может отправить сообщение своему маршрутизатору о желании присоединиться к группе. Помимо этого, надо сконфигурировать сетевой интерфейс хоста так, чтобы он стал захватывать из локальной сети кадры, несущие в себе пакеты группового вещания для той группы, к которой присоединился хост. Для этого необходимо настроить интерфейс на прослушивание определенного группового адреса канального уровня, соответствующего групповому IP-адресу. К сожалению, адресное пространство групповых IP-адресов в 32 раза объемнее пространства групповых MAC-адресов. То есть отображение этих двух адресных пространств друг на друга оказывается далеко не однозначным — на один и тот же групповой MAC-адрес отображается целый блок из 32 различных групповых IP-адресов. Следовательно, когда сетевой адаптер захватывает кадр, содержащий пакет группового вещания, существует значительная вероятность того, что этот пакет был направлен совсем другой группе. Однако эта ошибка скоро обнаруживается. Когда кадр передается вверх по стеку, протокол IP проверяет, совпадает ли групповой IP-адрес в поле адреса назначения инкапсулированного пакета с групповым IP-адресом данного интерфейса. (Отметим, что ни групповые IP-адреса, ни групповые MAC-адреса никогда не используются в качестве адресов отправителя.)

Принципы маршрутизации трафика группового вещания

Среди принципов маршрутизации трафика группового вещания можно отметить:

- маршрутизацию на основе доменов;
- учет плотности получателей группового трафика;
- два подхода к построению маршрутного дерева;
- концепцию продвижения по реверсивному пути.

Маршрутизация на основе доменов. Значительный объем хранимой и передаваемой по сети служебной информации, используемой для поддержания группового вещания, стал фактором, ограничивающим масштабируемость данной технологии. Для улучшения масштабируемости разработчики технологии группового вещания предложили традиционный для Интернета иерархический подход, основанный на доменах. Подобно автономным системам (доменам маршрутизации) и DNS-доменам вводятся **домены группового вещания**. Для доставки информации в пределах домена предлагаются одни методы и протоколы маршрутизации группового вещания, называемые *внутридоменными*, а в пределах многодоменной структуры — другие, называемые *междоменными*. Мы ограничимся в этом учебнике описанием средств продвижения пакетов группового вещания в пределах отдельного домена.

Учет плотности получателей группового трафика. Внутридоменные протоколы маршрутизации разделяются на два принципиально отличных класса:

- Протоколы **плотного режима** (Dense Mode, DM) разработаны в предположении, что в сетевом домене существует большое число принимающих узлов. Отсюда следует главная идея этих протоколов: сначала «затопить» сеть пакетами группового вещания по всем направлениям, останавливая продвижение пакетов, лишь когда находящийся на пути распространения трафика маршрутизатор явно сообщит, что далее ниже по потоку членов данной группы нет.
- Протоколы **разряженного режима** (Sparse Mode, SM) рассчитаны на работу в сети, в которой количество маршрутизаторов с подключенными к ним членами групп невелико по сравнению с общим числом маршрутизаторов. В такой ситуации выгоднее не усекать некоторые пути распространения широковещательной рассылки, а использовать явные сообщения о необходимости присоединения подсетей к дереву рассылки.

В сети, использующей протокол класса SM, необходимо существование центрального элемента, обычно называемого **точкой рандеву**, или **встречи** (Rendezvous Point, RP). Точка встречи должна существовать для каждой имеющейся в сети группы и быть единственной для группы. Все узлы, заинтересованные в получении информации, предназначенной той или иной группе, должны регистрироваться в соответствующей точке встречи. Функции точки (или нескольких точек) встречи выполняет специально назначенный для этого маршрутизатор. В сети может быть несколько маршрутизаторов, играющих роли точек встречи.

Два подхода к построению маршрутного дерева. Как и при решении задачи маршрутизации на основе индивидуальных адресов, в сети с групповым вещанием маршрутизаторы анализируют топологию сети, пытаясь найти кратчайшие пути доставки данных от источников к получателям. При этом все протоколы маршрутизации группового вещания используют один из следующих двух подходов.

- Для всех источников данной группы строится *единственный* граф связей, называемый **разделяемым деревом**. Этот граф связывает всех членов данной группы (точнее, все маршрутизаторы, к которым подключены локальные сети, имеющие в своем составе членов данной группы). Разделяемое дерево может включать также и необходимые для обеспечения связности маршрутизаторы, не имеющие в своих присоединенных сетях членов данной группы. Разделяемое дерево служит для доставки трафика всем членам данной группы от *каждого* из источников, вещающих на данную группу.
- Для каждой группы строятся *несколько* графов по числу источников, вещающих на каждую из этих групп. Каждый такой граф, называемый **деревом с вершиной в источнике**, служит для доставки трафика всем членам группы, но только от *одного* источника.

Концепция продвижения по реверсивному пути. Механизм, используемый для маршрутизации трафика группового вещания, в определенном аспекте является прямо противоположным (реверсивным) традиционному способу маршрутизации на основе индивидуальных адресов, при котором маршрутизаторы перемещают пакет по сети в направлении приемника. Напротив, все пакеты с групповым адресом маршрутизаторы тиражируют и передают копии во все стороны — на все интерфейсы, кроме того, с которого этот пакет поступил.

При этом в сложных сетях возможно образование петель. Для правильной работы сети зациклившиеся пакеты необходимо распознавать и отбрасывать. Петля не может возникнуть, если пакет прибыл от источника по ожидаемому пути, проложенному в соответствии с обычным алгоритмом маршрутизации, основанном на анализе таблиц маршрутизации. А именно — маршрутизатор проверяет, является ли входной интерфейс, получивший групповой пакет, интерфейсом, через который пролегает кратчайший путь к источнику. Он делает это с помощью обычной таблицы маршрутизации, которая, как известно, содержит указания о рациональных путях ко всем сетям составной интерсети. Проверка факта выполнения данного условия называется **продвижением по реверсивному пути** (Reverse Path Forwarding, RPF). Такое название объясняется тем, что эта процедура связана не столько с путями, ведущими вперед от текущего места нахождения пакета к пункту назначения, сколько с обратным (реверсивным) путем, который уже пройден пакетом от того места, где он находится сейчас, до источника. Только пакеты, которые прошли RPF-проверку, являются кандидатами для дальнейшего продвижения вдоль путей, ведущих к потенциальным получателям трафика группового вещания.

Концепция продвижения по реверсивному пути является главной при маршрутизации группового трафика независимо от того, какой протокол при этом использован. Механизм RPF применятся и в других вариантах организации группового вещания. Например, когда маршрутизатор пытается продвигать пакеты к точке встречи в сети, работающей в разряженном режиме, он выбирает интерфейс, от которого проходит кратчайший путь к точке встречи.

На этом этапе мы не предъявляли специфических требований к таблицам маршрутизации, на основании которых выполняется RPF-проверка. Некоторые протоколы, такие как DVMRP, строят собственную таблицу маршрутизации, в то время как, например, протокол PIM работает с таблицами маршрутизации, построенными другими протоколами.

Протоколы маршрутизации группового вещания

Протоколы маршрутизации группового вещания, такие как DVMRP, MOSPF и PIM, опираются на разные подходы, но в конечном итоге все они сводятся к построению покрывающего дерева, связывающего все хосты в определенной группе. Протоколы маршрутизации осуществляют постоянный мониторинг покрывающего дерева и время от времени отсекают ветви дерева, которые из-за изменения состояния сети уже не ведут к членам той или иной группы.

Дистанционно-векторный протокол маршрутизации группового вещания (Distance Vector Multicast Routing Protocol, DVMRP) был одним из первых протоколов продвижения группового трафика в исследовательской сети MBone. С самых общих позиций его можно охарактеризовать следующим образом:

- как следует из его названия, он основан на *дистанционно-векторном алгоритме* и, следовательно, обладает всеми особенностями, свойственными данному алгоритму;

- относится к классу *протоколов плотного режима*, использующих проверку *продвижения по реверсивному пути*;
- продвигает пакеты на основе *деревьев с вершинами в источниках*;
- является *протокольно зависимым* в том смысле, что для принятия решений о продвижении пакетов он не может использовать обычные (для индивидуальной рассылки) таблицы маршрутизации.

Главным недостатком протоколов плотного режима, к которым относится DVMRP, является то, что информация состояния для каждого источника должна храниться в каждом маршрутизаторе сети независимо от того, существуют члены групп вниз по потоку или нет. Если группа населена не очень плотно, то в сети нужно хранить значительный объем информации состояния и значительная часть пропускной способности может тратиться впустую.

Этот недостаток и стал толчком к разработке нового класса протоколов, названных протоколами *разряженного режима*, к которым, в частности, относятся протоколы MOSPF и PIM-SM. Вместо ориентации на существование большого количества членов группы протоколы разряженного режима подразумевают наличие их в небольшом количестве, причем рассеянном по сети, как это часто и бывает в действительности.

Протокол **MOSPF** (Multicast extensions to OSPF — расширения протокола OSPF для группового вещания) для поддержки группового вещания опирается на обычные механизмы OSPF. Маршрутизаторы MOSPF добавляют к информации о состоянии связей, распространяемой по протоколу OSPF, данные о членстве в группах узлов в непосредственно присоединенных сетях. Эти данные рассылаются по сети в дополнительном сообщении о членстве в группе (group membership). В результате помимо топологии связей маршрутизаторам MOSPF становится известно о наличии членов каждой из групп в каждой подсети области. На основании этой информации маршрутизатор находит дерево кратчайших путей для каждой группы. Это позволяет распространять групповые пакеты не широковещательно, а по кратчайшим путям от источника до подсетей, в которых есть активные члены группы.

Для получения данных о том, в какие группы входят конечные узлы в связанных с ним подсетях, маршрутизатор MOSPF использует запросы и ответы протокола IGMP. При каждом подключении узла к группе или исключении узла из группы маршрутизатор рассылает по сети новое сообщение о членстве в группе, так что можно считать, что протокол MOSPF задействует механизм явных уведомлений об изменении состава групп и поэтому относится к группе протоколов разряженного режима. Кроме того, известные положительные свойства протокола OSPF: устойчивое поведение при изменениях топологии сети, меньшие объемы служебного трафика по сравнению с протоколом RIP, а также возможность деления сети на области — полностью наследуются протоколом MOSPF, что делает его весьма привлекательным для применения в больших сетях.

Протокол PIM-SM является одной из двух версий протокола **PIM** (Protocol Independent Multicast — независимое от протокола групповое вещание):

- версии плотного режима **PIM-DM** (Protocol Independent Multicast — Dense Mode);
- версии разряженного режима **PIM-SM** (Protocol Independent Multicast — Sparse Mode).

Эти версии существенно отличаются друг от друга способом построения и использования покрывающего дерева, но у них есть и одно общее свойство. Оно вынесено в название каждого из этих протоколов и означает независимость данного протокола от конкретных протоко-

лов маршрутизации. Если DVMRP использует в своей работе механизмы RIP, а протокол MOSPF является расширением протокола OSPF, то протокол PIM может работать совместно с любым протоколом маршрутизации. Протокол PIM задействует готовые таблицы маршрутизации для продвижения групповых пакетов и служебных сообщений, и для него не имеет значения, с помощью какого протокола маршрутизации строятся эти таблицы.

Протокол PIM-DM похож на протокол DVMRP. Он, также являясь протоколом *плотного режима*, строит для доставки групповых пакетов *деревья с вершиной в источнике*, используя для этого проверки *продвижения по реверсивному пути* и технику *широковещания и усечения*. Основное отличие состоит в том, что протокол PIM-DM применяет готовую таблицу маршрутизации, а не строит ее сам, как это делает DVMRP.

Главной особенностью протокола PIM-SM является то, что он рассчитан на работу в *разряженном режиме*, то есть он посылает групповые пакеты только по явному запросу получателя. Для доставки данных каждой конкретной группе получателей протокол PIM-SM строит одно *разделяемое дерево*, общее для всех источников этой группы (рис. 17.11).

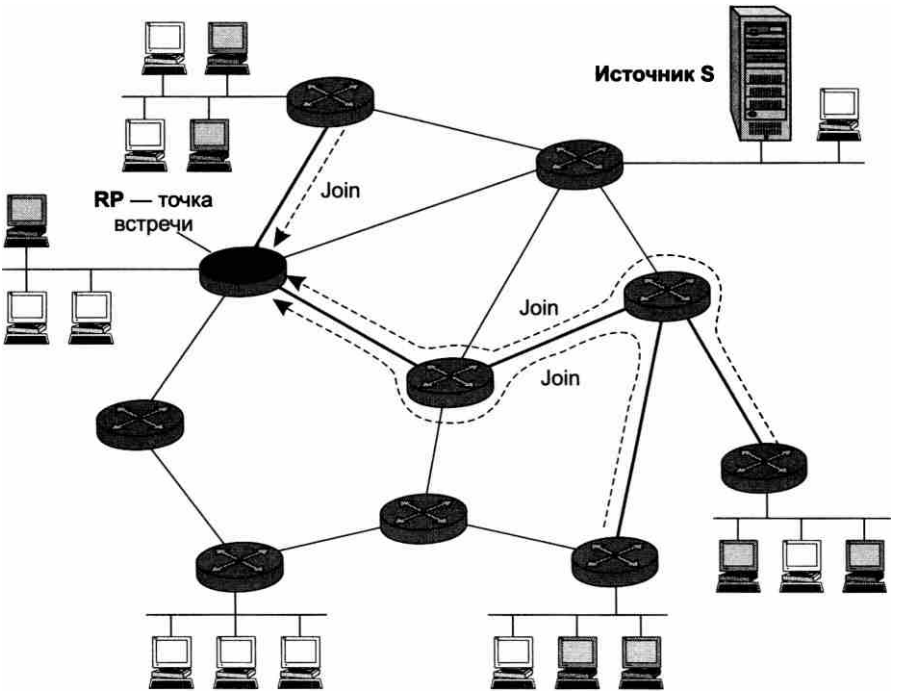


Рис. 17.11. Разделяемое дерево протокола PIM-SM

Вершина разделяемого дерева не может располагаться в источнике, так как источников может быть несколько. В качестве вершины разделяемого дерева используется специально выделенный для этой цели маршрутизатор, выполняющий функции *точки встречи* (RP). Все маршрутизаторы в пределах домена PIM-SM должны обладать согласованной информацией о расположении точки встречи. Различные группы могут иметь как одну и ту же, так и разные точки встречи.

Самым распространенным и, возможно, самым простым способом конфигурирования локальных (в пределах одного домена PIM-SM) точек встречи является назначение их *статически* среди множества маршрутизаторов данного домена. Это приводит к весьма определенной конфигурации и позволяет в дальнейшем легче находить ошибки, чем при других подходах.

Для получателей каждой конкретной группы и источников, вещающих на эту группу, маршрутизатор точки встречи является посредником, который связывает их между собой.

Процесс доставки протоколом PIM-SM группового трафика от источника к получателям, принадлежащим некоторой группе, может быть представлен трехэтапным:

- построение разделяемого дерева с вершиной в точке встречи, которое описывает пути доставки групповых пакетов между точкой встречи и членами данной группы. Это дерево называют также **деревом точки встречи** (Rendezvous Point Tree, RPT);
- построение **дерева кратчайшего пути** (Shortest Path Tree, SPT), которое будет доставлять пакеты между источником данной группы и точкой встречи;
- построение *набора* SPT-деревьев, которые ради повышения эффективности будут использованы для доставки пакетов непосредственно между источником и каждым из получателей группы.

ПРИМЕЧАНИЕ

Очередность этапов не фиксирована. Например, источники группового вещания могут начать передачу до того, как появятся слушатели, заинтересованные в этом трафике, или дерево кратчайшего пути между источником и его слушателями может уже быть построенным, когда будет сделан новый запрос на присоединение к группе.

(S) *Междоменное групповое вещание*

Поддержка QoS в машрутизаторах

Технологии стека TCP/IP были разработаны для эластичного трафика, который достаточно терпим к задержкам и вариациям задержек пакетов. Поэтому основное внимание разработчиков протоколов TCP/IP было сосредоточено на обеспечении надежной передачи трафика с помощью TCP. Однако со временем для борьбы с перегрузками на медленных линиях доступа в IP-маршрутизаторы были встроены многие механизмы QoS, в том числе механизмы приоритетных и взвешенных очередей, профилирования трафика и обратной связи. Однако эти механизмы использовались каждым сетевым администратором по своему усмотрению, без какой-либо стройной системы. И только в середине 90-х годов начались работы по созданию стандартов QoS для IP-сетей, на основе которых можно было бы создать систему поддержки параметров QoS в масштабах составной сети и даже Интернета.

В результате были разработаны две системы стандартов QoS для IP-сетей:

- система **интегрированного обслуживания** (**Integrated Services**, **IntServ**) ориентирована на предоставление гарантий QoS для потоков *конечных пользователей* «из конца в конец»;
- система **дифференцированного обслуживания** (**Differentiated Services**, **DiffServ**) предоставляет гарантии QoS в агрегированной форме *для классов трафика*.

Обе системы включают в себя все базовые элементы поддержки QoS:

- кондиционирование трафика;
- сигнализация, обеспечивающая координацию маршрутизаторов;
- резервирование пропускной способности интерфейсов маршрутизаторов для потоков и классов;
- приоритетные и взвешенные очереди.

Система интегрированного обслуживания

Система **IntServ** начала разрабатываться в IETF в начале 90-х годов, она была первой моделью, в рамках которой проблема обеспечения параметров QoS в сетях TCP/IP начала решаться систематически. Модель IntServ предполагает интегрированное взаимодействие маршрутизаторов сети по обеспечению требуемого качества обслуживания *вдоль всего пути потока* между конечными компьютерами.

Ресурсы маршрутизаторов (пропускная способность интерфейсов, размеры буферов) распределяются в соответствии с QoS-запросами приложений в пределах, разрешенных политикой QoS для данной сети. Эти запросы распространяются по сети сигнальным **протоколом резервирования ресурсов (Resource reSerVation Protocol, RSVP)**. Этот протокол подобен *сигнальным протоколам телефонных сетей*, с помощью которых вызывающий абонент сети запрашивает соединение с вызываемым абонентом. Однако специфика дейтаграммных пакетных сетей, естественно, накладывает свой отпечаток. Так, с помощью RSVP соединение в сети не устанавливается, так как IP-пакеты в любом случае (при резервировании или без него) будут передаваться маршрутизаторами между конечными узлами на основе записей таблиц маршрутизации. Резервируется же пропускная способность и запрашиваются желаемые параметры QoS для некоторого потока между двумя конечными узлами сети. Подобное резервирование является *однонаправленным*, так что если гарантированное качество обслуживания и пропускная способность должны быть обеспечены для двустороннего обмена, потребуются две операции резервирования, при этом параметры резервирования могут быть разными для каждого направления.

Протокол RSVP обеспечивает резервирование соединений как с двухточечной, так и с древовидной топологией. Древовидная топология соединения соответствует случаю передачи пакетов с групповым IP-адресом, когда один узел передает данные сразу нескольким получателям и копии пакета распространяются вдоль ветвей древовидного маршрута. Мультимедийные приложения, которым требуются гарантии пропускной способности и параметров QoS и которые, следовательно, являются одним из потенциальных пользователей протокола RSVP, могут задействовать групповое вещание для рационального расходования ресурсов сети. Поэтому разработчики протокола RSVP не могли не учесть этого варианта топологии соединений.

Рассмотрим работу протокола RSVP на примере сети, показанной на рис. 17.12. В этой сети имеется маршрут группового вещания, соединяющий передающий узел C1 с двумя принимающими узлами C2 и C3. Рассмотрим основные этапы процесса резервирования для нашего примера.

1. Источник данных (компьютер C1) посылает получателям специальное сообщение **PATH** по групповому адресу. В этом сообщении источник указывает параметры, рекоменду-

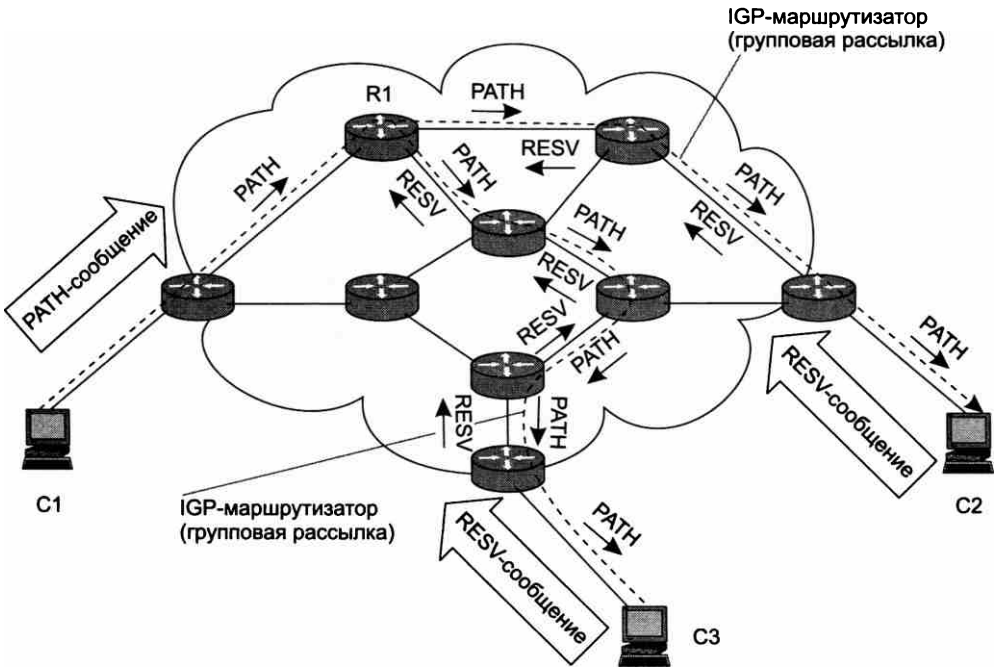


Рис. 17.12. Резервирование ресурсов по протоколу RSVP

емые для качественного приема трафика, который он будет отправлять по данному маршруту: верхние и нижние границы пропускной способности, а также максимальные значения задержки и вариации задержки. Эти параметры составляют **спецификацию трафика источника**. Сообщение PATH передается маршрутизаторами сети в направлении ко всем указанным в групповом адресе получателям. В качестве параметров трафика применяются параметры алгоритма ведра маркеров, то есть средняя скорость и глубина ведра.

2. Каждый маршрутизатор, поддерживающий протокол RSVP, получив сообщение PATH, фиксирует «состояние пути», которое включает предыдущий адрес источника сообщения, то есть последний по времени шаг в обратном направлении (ведущий к источнику). Это необходимо для того, чтобы ответ приемника прошел по тому же пути, что и сообщение PATH. Заметим, что резервирования ресурсов маршрутизатора пока не происходит.
3. После получения сообщения PATH каждый приемник (то есть компьютеры C2 и C3) отправляет в обратном направлении маршрутизатору, от которого он получил это сообщение, **запрос на резервирование ресурсов**, то есть сообщение **RESV**. В этом сообщении содержится **спецификация запроса приемника на резервирование**, в которой указываются нужные *приемнику* параметры требуемой пропускной способности и качества обслуживания. Эта спецификация вырабатывается приемником *на основе* спецификации трафика источника, но она не обязательно повторяет указанные источником параметры. Приемник использует спецификацию источника только как рекомендацию, но окончательное решение о том, какую пропускную способность и какие параметры качества обслуживания нужно резервировать, принимает приемник. Например, приемник может

решить принимать только аудиопоток от источника, а видео не принимать, тогда ему не потребуется резервировать пропускную способность, учитывающую передачу обоих потоков. Вместе со спецификацией запроса на резервирование приемник помещает в сообщение **спецификацию фильтра**, которая и определяет, к каким пакетам сеанса нужно применять данное резервирование (например, по типу транспортного протокола и номеру порта он может выделить только аудиопакеты). Вместе спецификации запроса и фильтра представляют собой **дескриптор потока**, для которого выполняется резервирование.

4. Каждый маршрутизатор, получив сообщение RESV, проверяет, во-первых, имеются ли у маршрутизатора ресурсы, необходимые для поддержания запрашиваемой пропускной способности и уровня QoS, а во-вторых, имеет ли пользователь право на резервирование ресурсов. Если запрос не может быть удовлетворен (из-за недостатка ресурсов или ошибки авторизации), маршрутизатор возвращает сообщение об ошибке отправителю. Если запрос принимается, то маршрутизатор посылает сообщение RESV далее вдоль маршрута следующему маршрутизатору, а данные о требуемом уровне QoS передаются тем механизмам маршрутизатора, которые ответственны за управлением трафиком.
5. Протокол RSVP не определяет способ, с помощью которого маршрутизатор проверяет, достаточно ли у него ресурсов для принятия запроса на резервирование. Предполагается, что такая проверка может быть реализована программным обеспечением маршрутизатора, а ее детали определяются производителем маршрутизатора индивидуально. Например, если у маршрутизатора сконфигурирована очередь для обслуживания приоритетного трафика, он может вести учет выделения пропускной способности этой очереди различным потокам и при поступлении очередного запроса RESV сравнивать наличие свободной пропускной способности с запрашиваемой. При положительном результате проверки маршрутизатор запоминает новые параметры резервирования и вычитает их из счетчиков соответствующих свободных ресурсов.
6. Когда последний в обратном направлении маршрутизатор получает сообщение RESV и принимает запрос, то он посылает подтверждающее сообщение узлу-источнику. При групповом резервировании учитывается тот факт, что в точках разветвления дерева доставки несколько резервируемых потоков сливаются в один. Так, в маршрутизаторе R1 в рассматриваемом примере сливаются сообщения RESV от приемников C2 и C3. Если для всех резервируемых потоков запрашивается одинаковая пропускная способность, то она требуется и для общего потока, а если запрашиваются различные величины пропускной способности, то для общего потока выбирается максимальная.
7. После установления состояния резервирования в сети источник начинает отправлять данные, которые обслуживаются на всем пути к приемнику (приемникам) с заданным качеством обслуживания.

Для того чтобы параметры резервирования можно было применить затем к трафику данных, необходимо, чтобы сообщения RSVP и пакеты данных следовали через сеть *одним и тем же маршрутом*. Это можно обеспечить, если передавать сообщения RSVP на основе тех же записей таблиц маршрутизации, которые применяются для пользовательского трафика.

Резервирование можно отменить прямо или косвенно. Прямая отмена выполняется по инициативе источника или приемника с помощью соответствующих сообщений протокола RSVP. Неявная отмена происходит по тайм-ауту: состояние резервирования имеет срок

жизни, как, например, и динамические записи в таблицах маршрутизации, и приемник по протоколу RSVP должен периодически подтверждать резервирование. Если же подтверждающие сообщения перестают поступать, то резервирование отменяется по истечении его срока жизни. Такое резервирование называется мягким.

Модель резервирования ресурсов IntServ, опирающаяся на протокол RSVP, не нашла широкого применения. Основной причиной было опасение поставщиков услуг Интернета за работоспособность своих маршрутизаторов. Понятно, что применение модели IntSrerv в масштабах Интернета привело бы к необходимости хранения маршрутизаторами информации о состоянии резервирования для миллионов отдельных пользовательских потоков, и это намного больше, чем требует сегодняшняя модель, в соответствии с которой маршрутизаторы хранят данные только об агрегированных префиксах сетей назначения (даже эта намного более экономная модель привела в середине 2010-х годов к необходимости хранения магистральными маршрутизаторами Интернета более полумиллиона записей в таблицах маршрутизации).

Другим недостатком модели IntServ является игнорирование ею структуры Интернета, состоящей из сетей различных провайдеров. Запрос на резервирование должен пройти в общем случае через сети различных провайдеров, при этом каждый провайдер должен согласиться с тем, что его маршрутизаторы должны выделять часть своих ресурсов потоку данных неизвестного пользователя. Модель IntServ игнорирует эту проблему, не предлагая механизмов аутентификации пользователей и политики выделения ресурсов, а без этого механизмом резервирования легко могут воспользоваться злоумышленники, что приведет к исчерпанию ресурсов сети.

Для преодоления этих недостатков была разработана другая модель, названная моделью дифференцированного обслуживания.

Система дифференцированного обслуживания

Дифференцированное обслуживание (DiffServ) опирается на те же механизмы QoS, что и интегрированное обслуживание, однако в качестве объектов обслуживания рассматриваются не отдельные потоки, а классы трафика.

Классом трафика называется совокупность поступающих на обработку пакетов, обладающих общими признаками, например все пакеты голосовых приложений или все пакеты с MTU в определенных пределах.

В отличие от потока, в классах трафика пакеты не различаются по их маршрутам; это отличие иллюстрирует рис. 17.13. Так, маршрутизатор R1 относит все пакеты, например пакеты, требующие приоритетного обслуживания, к одному классу. Для интерфейса i1 они представлены одним и тем же входящим агрегированным потоком, несмотря на то что в этот поток входят пакеты с разными маршрутами: как пакеты, направляемые через маршрутизатор R3, так и пакеты, направляемые через маршрутизатор R2. Так как пакеты приоритетного класса направляются маршрутизатором по разным маршрутам, то агрегированный поток пакетов данного класса разделяется на два агрегированных потока. Маршрутизатор R2 оперирует уже другим составом приоритетного класса, поскольку в него вошли не все потоки интерфейса i1 маршрутизатора R1.

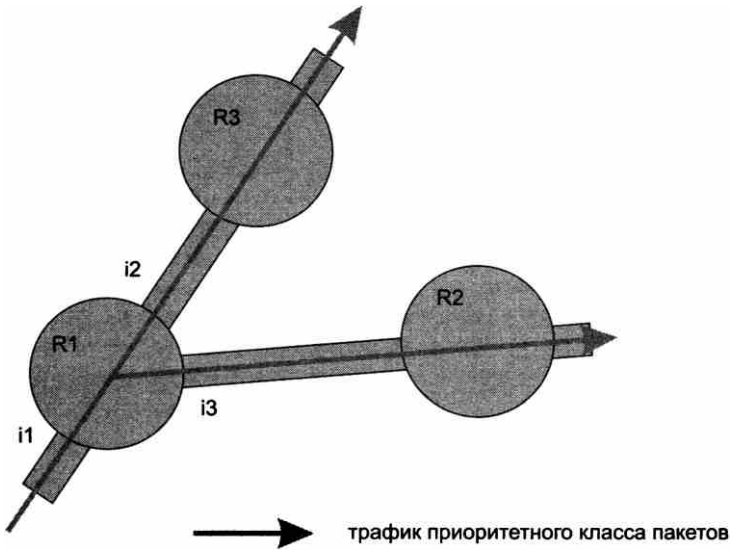


Рис. 17.13. Передача класса трафика маршрутизаторами

Таким образом, в агрегированный поток некоторого класса входит некоторое (возможно большое) число пользовательских потоков, иногда называемых микропотоками, подчеркивая их отличие от агрегированного потока.

Предполагается, что в класс объединяются пользовательские потоки, которые имеют близкие требования к качеству облуживания — задержкам пакетов и вариациям задержек. Требования к пропускной способности для каждого индивидуального потока в этой модели не обеспечиваются, здесь можно говорить только о пропускной способности для класса в целом.

Обычно в сети DiffServ поддерживается дифференцированное обслуживание небольшого количества классов трафика, например двух (чувствительного к задержкам и эластичного) или трех (к первым двум прибавляется класс, требующий гарантированной доставки пакетов с определенным минимумом скорости трафика). Небольшое количество классов определяет масштабируемость этой модели, так как маршрутизаторы не должны запоминать состояния каждого пользовательского потока. Высокая степень масштабирумости Diffserv обеспечивается также тем, что каждый маршрутизатор самостоятельно принимает решение о том, как он должен обслуживать тот или иной класс трафика, не согласуя свои действия с другими маршрутизаторами. Такой подход назван *независимым поведением маршрутизаторов* (Per Hop Behavior, PHB). Так как в модели DiffServ маршруты пакетов не отслеживаются, то здесь не используется сигнальный протокол резервирования ресурсов, подобный протоколу RSVP в модели IntServ. Вместо этого маршрутизаторы сети выполняют статическое резервирование ресурсов для каждого из поддерживаемых сетью классов. Например, если маршрутизатор поддерживает несколько очередей с алгоритмом взвешенного обслуживания и за каждой очередью закреплен определенный класс трафика, то, выделяя очередям некоторый процент пропускной способности выходного интерфейса маршрутизатора, мы тем самым резервируем пропускную способность для классов трафика.

В качестве признака принадлежности IP-пакета к определенному классу в DiffServ используется метка, переносимая в поле приоритета IP-пакета (байт ToS), которое с появлением стандартов DiffServ было переопределено и названо байтом DS. Эта метка должна переносить между маршрутизаторами информацию о принадлежности пакета к определенному классу.

Как показано на рис. 17.14, байт DS переопределяет значения битов байта ToS, определенных ранее в соответствующих спецификациях (RFC 791, RFC 1122, RFC 1349).

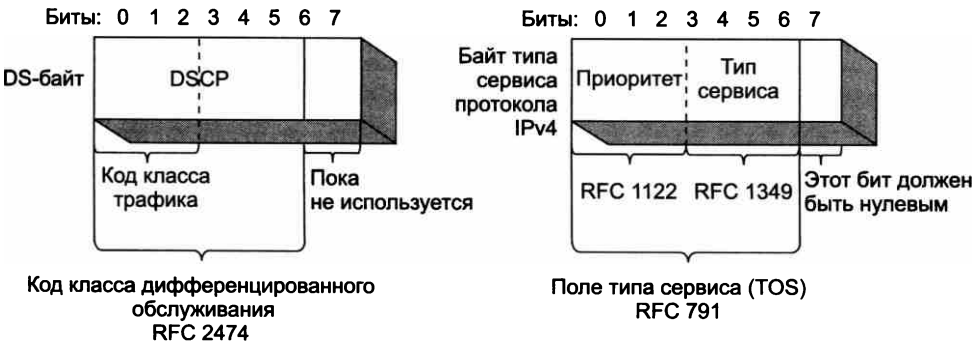


Рис. 17.14. Соответствие битов байта DS битам поля типа сервиса

В настоящее время используются только старшие шесть битов байта DS, причем из них только старшие три требуются для определения класса трафика (что дает не более восьми различных классов). Младший бит (из используемых шести) байта DS обычно переносит признак IN — индикатор того, что пакет «вышел» из профиля трафика и его можно отбросить или перевести в другой класс. Промежуточные два бита обычно описывают различные варианты обслуживания пакетов внутри одного класса трафика.

Маршрутизатор, поддерживающий модель DiffServ, должен обеспечивать классификацию, маркирование, измерение и кондиционирование трафика, его обслуживание в приоритетной или взвешенной очереди и сглаживание.

Хотя маркировкой пакетов может заниматься каждый маршрутизатор сети, в модели дифференцированного обслуживания основным вариантом считается маркировка пакетов на границе сети, поддерживающей эту модель и находящейся под административным контролем одной организации. Такая сеть называется DiffServ-доменом. При выходе пакетов за пределы DiffServ-домена маркировка снимается, так что другой домен может назначить ее заново. Пограничные маршрутизаторы DiffServ-домена исполняют роль контрольно-пропускных пунктов домена, проверяя входящий трафик и определяя, имеет ли он право на дифференцированное обслуживание.

Модель DiffServ подразумевает существование соглашения об уровне обслуживания (SLA) между доменами с общей границей. Это соглашение определяет критерии политики предоставления сервиса, профиль трафика, а также гарантируемые параметры QoS. Ожидается, что трафик будет формироваться и сглаживаться в выходных точках домена в соответствии с SLA, а во входной точке домена кондиционироваться в соответствии с правилами политики. Любой трафик «вне профиля» (например, выходящий за верхние границы полосы пропускания, указанной в SLA) не получает гарантий обслуживания (или

же оплачивается по повышенной стоимости в соответствии с SLA). Правила политики предоставления сервиса могут включать время дня, адреса источника и приемника, транспортный протокол, номера портов. В том случае, когда соблюдаются правила политики и трафик удовлетворяет оговоренному профилю, DiffServ-домен должен обеспечить при обслуживании этого трафика параметры QoS, зафиксированные в SLA.

На сегодняшний день в IETF разработано два стандарта пошагового продвижения пакетов для схемы PHB, которые представляют два разных варианта обслуживания.

- *Быстрое продвижение* (Expedited Forwarding, EF) характеризуется значением кода 10111 в байте DS (десятичное значение 46) и представляет собой высший уровень качества обслуживания, обеспечивая минимум задержек и вариаций задержек.
- *Гарантированная доставка* (Assured Forwarding, AF) характеризуется четырьмя классами трафика и тремя уровнями отбрасывания пакетов в каждом классе — всего получается 12 различных вариантов трафика. Каждому классу трафика выделяется определенные минимум пропускной способности и размер буфера для хранения его очереди. Трафик, параметры которого превышают указанные в профиле, доставляется с меньшей степенью вероятности, чем трафик, удовлетворяющий условиям профиля. Это означает, что качество его обслуживания может быть понижено, но он не обязательно будет отброшен.

На основе этих **пошаговых спецификаций** и соответствующих соглашений об уровне обслуживания (SLA) могут быть построены **сервисы** для конечных пользователей «из конца в конец» — это EF- и AF-сервис соответственно. Предполагается, что провайдер наряду с EF- и AF-услугами предоставляет и стандартные услуги IP-сети, то есть услуги «по возможности» без гарантий пропускной способности и параметров QoS.

Основное назначение EF-сервиса — обеспечение качества обслуживания, сопоставимого с качеством обслуживания выделенных каналов, поэтому этот сервис называется также *сервисом виртуальных выделенных каналов*.

Поскольку EF-сервис допускает полное вытеснение другого трафика (например, при реализации EF-сервиса с помощью приоритетной очереди), то его реализация должна включать некоторые средства ограничения влияния EF-трафика на другие классы трафика, например путем ограничения скорости EF-трафика на входе маршрутизатора по алгоритму ведра маркеров. Максимальная скорость EF-трафика и, возможно, величина пульсаций должны устанавливаться сетевым администратором.

Четыре класса AF-сервиса ориентированы на гарантированную доставку, но без минимизации уровня задержек пакетов, как это оговорено для EF-сервиса. Гарантированная доставка выполняется в том случае, когда входная скорость трафика не превышает отведенной данному классу минимальной пропускной способности. Реализация классов AF-трафика хорошо сочетается с EF-сервисом — EF-трафик может обслуживаться по приоритетной схеме, но с ограничением интенсивности входного потока. Оставшаяся пропускная способность распределяется между классами AF-трафика в соответствии с алгоритмом взвешенного обслуживания, который обеспечивает необходимую пропускную способность, но не минимизацию задержек. Реализация AF-сервиса предполагает (но не требует) взвешенного обслуживания для каждого класса с резервированной полосой пропускания, а также применения обратной связи (в форме RED).

Относительная простота определяет недостатки дифференцированного обслуживания. Главным недостатком является сложность предоставления количественных гарантий

качества обслуживания «из конца в конец». Поясним это на примере сети, изображенной на рис. 17.15.

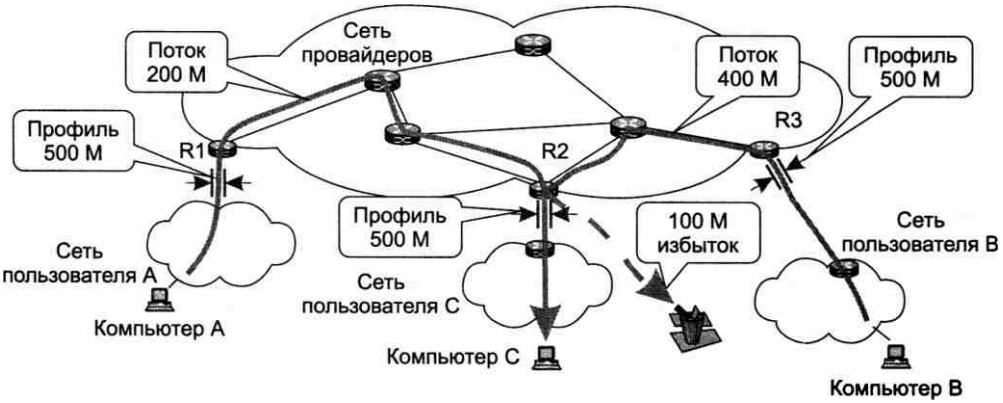


Рис. 17.15. Неопределенность уровня обслуживания в модели DiffServ

В этом примере сеть провайдера соединена с тремя сетями корпоративных пользователей: A, B и C. Сеть провайдера предоставляет пользователям услуги двух классов DiffServ: быстрого продвижения EF и обслуживания «с максимальными усилиями». У провайдера заключено соглашение SLA со своими клиентами, в котором он гарантирует услугу EF своим пользователям в том случае, если скорость такого трафика от клиента не превышает 500 Мбит/с. Трафик EF-услуги обслуживается маршрутизатрами провайдера с помощью приоритетных очередей. Для того чтобы приоритетный трафик клиентов оставил достаточно ресурсов маршрутизаторов для обслуживания трафика класса «с максимальными усилиями», провайдер применил профилирование трафика на интерфейсах, соединяющих его маршуртизаторы с маршрутизаторами сетей пользователей. Профилирование проводится с уровнем скорости трафика CIR = 500 Мбит/с, точно в соответствии с соглашениями SLA. Профилирование применяется в обоих направлениях, чтобы защитить как свою сеть, так и сети пользователей от избыточного (не предусмотренного SLA) приоритетного трафика.

Как видно из рисунка, такое профилирование не всегда работает так, как задумано. В нашем примере компьютер A посылает на компьютер C поток данных класса EF со средней скоростью 200 Мбит/с. Эта скорость ниже, чем оговоренный предел 500 Мбит/с, поэтому профилирующий элемент входного интерфейса маршрутизатора R1 пропускает все пакеты потока. Компьютер B также посылает поток данных класса EF компьютеру C, его скорость равна 400 Мбит/с, что также устраивает профилирующий элемент маршрутизатора R3.

Однако когда эти два потока поступают на интерфейс маршрутизатора R2, соединяющий его с сетью пользователя C, выясняется, что их суммарная скорость пакетов класса EF уже превышает порог профилирования, так как она становится равной 600 Мбит/с. Если профилирующий элемент маршрутизатора R2 настроен стандартным образом, то он просто будет отбрасывать пакеты, не удовлетворяющие условиям алгоритма ведра маркеров с параметром скорости 500 Мбит/с. Поток отброшенных пакетов будет иметь скорость 100 Мбит/с.

Провайдер может попытаться смягчить ситуацию, изменив конфигурацию профилирующего элемента маршрутизатора R2 и настроив его не на отбрасывание пакетов-нарушителей, а на маркирование их как пакетов класса «по возможности» и обслуживая их соответствующим образом. Однако это будет нарушением соглашения SLA, хотя каждый из пользователей его не нарушал — ведь и пользователь A, и пользователь B направляли трафик класса EF со скоростью, не превышающей 500 Мбит/с. Отказ от профилирования приоритетного трафика в направлении к пользовательским сетям также не является решением проблемы, так как может привести к полному вытеснению из сети трафика «по возможности» (например, при сложении многих потоков приоритетного трафика на одном из интерфейсов маршрутизатора).

Описанная проблема в рамках модели DiffServ неразрешима. Она является следствием того, что в этой модели нет процедуры проверки наличия ресурсов вдоль пути следования трафика, той процедуры, которую в модели IntServ выполняет протокол RSVP. Таким образом, мы видим, что, решив проблему масштабируемости (за счет агрегирования потоков данных в классы), разработчики модели DiffServ лишили систему обеспечения качества обслуживания определенности — ее работа зависит от того, насколько удачно в сети провайдера распределились потоки приоритетного трафика. Провайдер может смягчить эту проблему за счет мониторинга распределения трафика клиентов по различным маршрутам в пределах своей сети и соответствующего подбора скоростей интерфейсов и профилей.

Недостатки обеих моделей — IntServ и DiffServ — объясняют отсутствие услуг QoS, предоставляемых в масштабах всего Интернета. Отдельные провайдеры предлагают такие услуги, но только в пределах своей сети, где они могут применять как IntServ, так и отдельные механизмы QoS в соответствии с собственной нестандартной моделью.

(S) *Вопросы реализации маршрутизаторов. Функциональная схема. Операционная система Cisco IOS. Классификация маршрутизаторов*

Выводы

Протоколы маршрутизации генерируют для каждого маршрутизатора согласованные таблицы маршрутизации, которые позволяют обеспечить доставку пакета по рациональному маршруту от исходной сети в сеть назначения за конечное число шагов.

Адаптивная маршрутизация обеспечивает автоматическое обновление таблиц маршрутизации после изменения конфигурации сети.

Адаптивные протоколы маршрутизации делятся на дистанционно-векторные (например, RIP) и протоколы состояния связей (например, OSPF).

Дистанционно-векторные протоколы хорошо работают только в небольших сетях. В больших сетях они периодически засоряют линии связи интенсивным трафиком, к тому же изменения конфигурации не всегда корректно могут отражаться протоколами этого типа, так как маршрутизаторы не имеют точного представления о топологии связей в сети, а располагают только косвенной информацией — вектором расстояний.

Протоколы состояния связей обеспечивают каждый маршрутизатор информацией, достаточной для построения точного графа связей сети. Все маршрутизаторы работают на основании одного и того же графа, что делает процесс маршрутизации более устойчивым к изменениям конфигурации. Служебный трафик, создаваемый протоколами состояния связей, гораздо менее интенсивный, чем у дистанционно-векторных протоколов.

Протоколы маршрутизации Интернета делятся на внешние (EGP), которые переносят маршрутную информацию между автономными системами, и внутренние (IGP), применяемые только в пределах определенной автономной системы.

Пограничный (внешний) шлюзовой протокол BGP версии 4 является сегодня основным протоколом обмена маршрутной информацией между автономными системами Интернета. BGP успешно работает при любой топологии связей между автономными системами, что соответствует современному состоянию Интернета.

Групповое вещание представляет собой эффективный механизм доставки одной и той же информации от одного источника определенному кругу абонентов сети. Одним из основных элементов группового вещания является протокол IGMP, с помощью которого хосты сообщают о своем «желании» присоединиться к некоторой группе, а маршрутизатор узнает о принадлежности хостов к той или иной группе в непосредственно подключенных к нему подсетях. Важную функцию также выполняют протоколы маршрутизации группового вещания DVMRP, MOSPF и PIM, продвигающие через сеть произвольной конфигурации пакеты с информацией для групповых получателей.

Для поддержки QoS в IP-сетях разработаны две системы: система интегрированного обслуживания IntServ, ориентированная на предоставление гарантий QoS для потоков конечных пользователей «из конца в конец», и система дифференцированного обслуживания DiffServ, предоставляющая гарантии QoS в агрегированной форме для классов трафика.

Контрольные вопросы

1. Может ли работать маршрутизатор, не имея таблицы маршрутизации? Варианты ответов:
   а) может, если выполняется маршрутизация от источника;
   б) нет, это невозможно;
   в) может, если в маршрутизаторе задан маршрут по умолчанию;
   г) может, если выполняется лавинная маршрутизация.
2. Можно ли обойтись в сети без протоколов маршрутизации?
3. По какой причине в протоколе RIP расстояние в 16 хопов между сетями полагается недостижимым? Варианты ответов:
   а) поле, отведенное для хранения значения расстояния, имеет длину 4 двоичных разряда;
   б) сети, в которых работает RIP, редко бывают большими;
   в) для получения приемлемого времени сходимости алгоритма.
4. Какие параметры сети учитывают метрики, поддерживаемые протоколом OSPF? Варианты ответов:
   а) пропускная способность;
   б) количество хопов;
   в) надежность каналов связи.
5. Предложите варианты метрики, которая одновременно учитывает пропускную способность, надежность и задержку линий связи.

Часть V

Глобальные компьютерные сети

- Глава 18. Организация и услуги глобальных сетей
- Глава 19. Транспортные технологии глобальных сетей
- Глава 20. Технология MPLS
- Глава 21. Ethernet операторского класса
- Глава 22. Виртуальные частные сети

Технология IP, которую мы рассматривали в предыдущей части книги, позволяет строить составные сети различного типа, как локальные, так и глобальные. Протокол IP стал сегодня тем протоколом, который объединяет многочисленные сети операторов связи и предприятий в глобальную мировую компьютерную сеть, называемую Интернетом. Это привело к тому, что одной из основных услуг операторов связи, относящейся к транспортным услугам компьютерных сетей, стал доступ в Интернет, а операторы связи по совместительству стали поставщиками (провайдерами) услуг Интернета. Другим популярным типом услуг сетей операторов связи является услуга виртуальных частных сетей, которая позволяет объединить отдельные территориально рассредоточенные сети некоторого предприятия в единую корпоративную сеть.

Технология IP не является единственной технологией коммутации пакетов, которая работает в глобальных сетях. Типичная глобальная сеть имеет многоуровневую структуру, в которой IP занимает верхний уровень (если рассматривать только уровни, обеспечивающие транспортировку данных), а под уровнем IP работают пакетные технологии канального уровня. Для глобальных сетей был разработан ряд технологий, учитывающих особенности этого типа сетей, в частности X.25, Frame Relay, ATM и MPLS. Объединяет все перечисленные технологии то, что они основаны на технике виртуальных каналов. Основная причина успеха техники виртуальных каналов в глобальных сетях состоит в том, что она обеспечивает гораздо более высокую степень контроля над соединениями между пользователями сети и путями прохождения информационных потоков через узлы сети, чем дейтаграммная техника.

Опыт сосуществования IP с технологиями, основанными на механизме виртуальных каналов, привел в середине 90-х годов к появлению гибридной технологии MPLS, которая тесно интегрирована с протоколами стека IP, так что иногда ее называют IP/MPLS. При использовании MPLS протоколы маршрутизации стека TCP/IP служат для исследования топологии сети и нахождения рациональных маршрутов, а продвигаются пакеты на основе техники виртуальных каналов. Интеграция IP и MPLS оказалась очень удачной, так что эта комбинация в настоящее время вытеснила из глобальных сетей технологии Frame Relay и ATM. MPLS сегодня используется в различных качествах, и как внутренняя технология операторов связи, дающая высокую степень контроля над трафиком и обеспечивающая быстрое восстановление соединений, и как технология, на которой строятся услуги оператора связи.

Сравнительно недавно класс технологий глобальных сетей пополнился новым представителем — Ethernet операторского класса (Carrier Ethernet). Этим именем одновременно называют как транспортные услуги глобальных сетей, которые предоставляются пользователям с интерфейсом Ethernet, так и усовершенствованную версию классической технологии Ethernet, снабженную некоторыми новыми свойствами, необходимыми для успешной работы в глобальных сетях в качестве внутренней транспортной технологии провайдера (называемой в этом случае Carrier Ethernet Transport).

MPLS и Carrier Ethernet Transport служат основой для предоставления услуг виртуальных частных сетей (VPN). Этот вид услуг становится все более популярным, так как позволяет построить корпоративную сеть без необходимости создания или аренды физических каналов связи. Виртуальная частная сеть создается в IP/MPLS или IP/CET сети провайдера услуг и имитирует свойства частной сети, такие как изолированность от сетей других пользователей, предсказуемая производительность и безопасность.

Обеспечение высокоскоростного доступа к сетевой магистрали представляет собой сегодня масштабную и специфическую проблему. Действительно, скорость нужно повысить на миллионах линий связи, соединяющих помещения пользователей с ближайшими центральными офисами операторов связи. Поэтому традиционные для магистрали решения, основанные на прокладывании оптических кабелей к домам и офисным зданиям и применении в сети активного оборудования технологий SDH или OTN, для обеспечения массового доступа часто оказываются экономически неоправданными, хотя их популярность растет. Более эффективными оказываются технологии, в которых задействуется существующая кабельная инфраструктура (например, технология ADSL, работающая на абонентских окончаниях телефонной сети) или же оптические линии доступа используются как разделяемая среда, построенная на пассивных недорогих разветвителях сигнала (технология PON).

ГЛАВА 18 Организация и услуги глобальных сетей

Сети операторов связи

Сегодня глобальные компьютерные сети операторов связи являются движущей силой и местом приложения практически всех новых транспортных технологий компьютерных сетей. Корпоративные сети, как правило, уже не строятся на основе собственной инфраструктуры глобальных связей — теперь для объединения локальных сетей своих территориально разнесенных подразделений предприятия обращаются к транспортным услугам глобальных сетей операторов связи.

Специализированное предприятие, которое создает телекоммуникационную сеть для оказания общедоступных услуг, владеет этой сетью и поддерживает ее работу, называется **оператором связи** (telecommunication carrier).

Глобальные компьютерные сети операторов связи являются частью их телекоммуникационной сети, в которую также могут входить сети других типов: телефонная стационарная сеть, телефонная мобильная сеть, телевизионная сеть. С помощью этих сетей операторы связи оказывают широкий спектр услуг как конечным пользователям, так и друг другу.

Операторы связи отличаются друг от друга:

- набором предоставляемых услуг;
- территорией, в пределах которой предоставляются услуги;
- типом клиентов, на которых ориентированы их услуги;
- имеющейся во владении оператора инфраструктурой — линиями связи, коммутационным оборудованием, информационными серверами и т. п.;
- отношением к монополии на предоставление услуг.

Услуги операторов связи

Современные операторы связи обычно оказывают услуги нескольких типов — как правило, это услуги выделенных каналов, телефонии и компьютерных сетей. Все эти услуги могут быть сгруппированы и иерархически упорядочены. На рис. 18.1 фрагментарно показана взаимосвязь некоторых наиболее популярных современных телекоммуникационных услуг.

Услуги более высокого уровня опираются на услуги нижележащих уровней. *Услуги предоставления каналов связи в аренду* относятся к самому нижнему уровню, так как пользователю приходится при этом самостоятельно выполнять дополнительную работу — строить с помощью предоставленных каналов собственную сетевую инфраструктуру (устанавливать телефонные коммутаторы или коммутаторы и маршрутизаторы компьютерных сетей). Обычно к таким услугам обращаются другие операторы связи, у которых для построения своей сети нет собственных каналов связи.

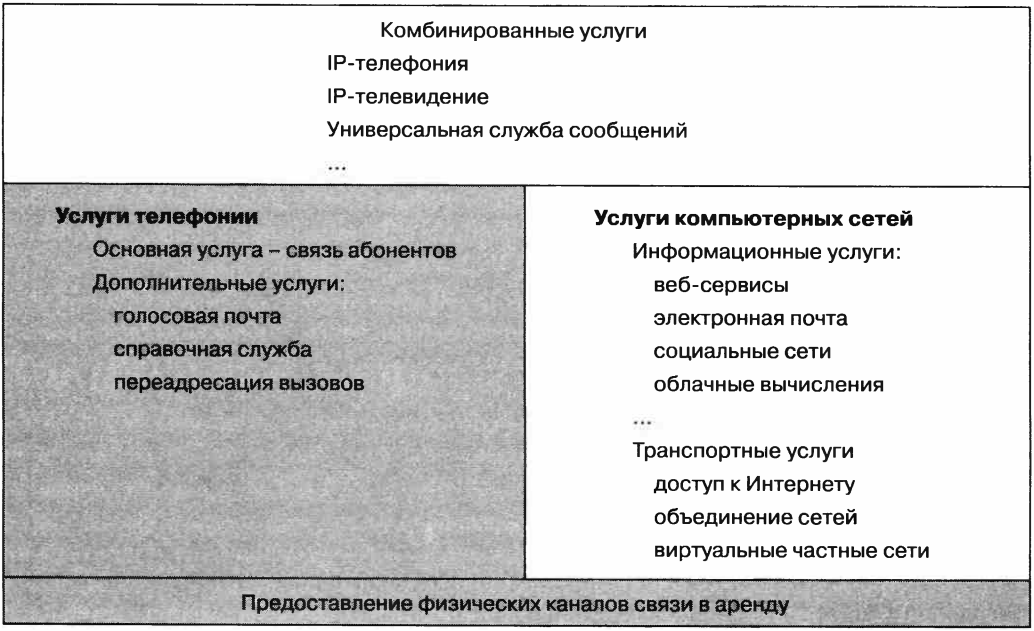


Рис. 18.1. Взаимосвязь услуг телекоммуникационной сети (закрашенные области соответствуют традиционным услугам операторов связи)

Следующий более высокий уровень составляют две большие группы услуг: услуги телефонии и услуги компьютерных сетей.

Услуги телефонии — это, прежде всего, всем нам знакомая телефонная связь абонентов. Однако с течением времени наряду с этой традиционной услугой операторы связи стали предлагать абонентам голосовую почту, справочную службу, переадресацию вызовов, блокирование определенных номеров, ограничение спама и другие вспомогательные сервисы.

Услуги компьютерных сетей стали предлагаться намного позже, чем телефонные, однако сейчас подавляющее большинство операторов связи предоставляют эти услуги. Они подразделяются на:

- *информационные* — веб-сервис, электронная почта, социальные сети и др.;
- *транспортные* — доступ в Интернет, создание виртуальных частных сетей.

Каждый из описанных уровней услуг может быть надстроен услугами более высокого уровня. Например, на основе простой услуги доступа в Интернет оператор может разработать и предложить клиентам в качестве более развитой услуги создание для них веб-порталов и размещение их в своей сети.

Верхний уровень сегодня занимают *комбинированные услуги*, реализация которых требует совместного использования компьютерных и телефонных сетей. Ярким примером такой услуги является международная IP-телефония, которая уже сейчас отобрала у традиционной международной телефонии значительную часть клиентов.

Операторы связи применяют различные транспортные технологии, например IP, Ethernet, OTN, SDH, DWDM, — эти технологии работают на разных уровнях стека протоколов сети,

обладают различными свойствами и могут работать в различных сочетаниях. Оператор связи использует эти технологии как для создания своих сетей, так и для предоставления услуг своим клиентам. Соотношение понятий *технология* и *услуга* можно пояснить следующими утверждениями:

- одна и та же технология может быть использована для предоставления различных услуг: например, технология IP может служить как для доступа в Интернет, так и для организации виртуальной частной сети;
- одна и та же услуга может быть реализована на основе разных технологий: например, виртуальную частную сеть можно построить на основе технологии IP, Ethernet и MPLS;
- имеются такие услуги, которые можно предоставлять на основе только какой-то одной специфической технологии: например, услугу выделенной волны можно предоставлять только на основе технологии DWDM.

Преобладающий тип услуг отражается в названиях телекоммуникационных компаний. Мы говорим «оператор» применительно к традиционным компаниям, основным бизнесом которых всегда были телефонные услуги и услуги предоставления каналов связи в аренду. Название «провайдер услуг», или «провайдер», стало популярным с массовым распространением Интернета и его информационных услуг.

Потребители услуг

Все множество клиентов — потребителей инфотелекоммуникационных услуг — можно разделить на два больших класса: массовые индивидуальные клиенты и корпоративные клиенты.

В первом случае местом потребления услуг выступает квартира или частный дом, а клиентами — жильцы, которым нужны прежде всего базовые услуги — телефонная связь, телевидение, радио, доступ в Интернет. Для **массовых индивидуальных клиентов** очень важна экономичность услуги — низкая месячная оплата, возможность использования стандартных терминальных устройств, таких как телефонные аппараты, телевизионные приемники, персональные компьютеры, а также возможность задействовать существующую кабельную систему между ближайшим офисом оператора связи и домом клиента. Присутствующая во многих местностях традиционная телефонная проводка — это серьезное ограничение для предоставления услуг доступа в Интернет и новых услуг компьютерных сетей, так как она не была рассчитана на передачу данных, а подведение к каждому дому нового качественного кабеля, например волоконно-оптического, — дело дорогое (хотя этот вариант становится все более доступным). Поэтому для предоставления компьютерных услуг таким клиентам разработаны специфические технологии доступа через существующие в доме окончания телефонной сети. В этом случае новые скоростные цифровые технологии доступа (DSL) используют телефонную сеть, но только на отрезке между домом клиента и офисом оператора связи, а далее данные передаются в обход телефонной сети по компьютерной сети с коммутацией пакетов. Существуют также технологии доступа, в которых для передачи данных применяется имеющаяся в городе сеть кабельного телевидения.

Корпоративные клиенты — это предприятия и организации различного профиля. Мелкие предприятия по набору требуемых услуг не слишком отличаются от массовых клиентов — это те же базовые телефония и телевидение, а также доступ к информационным ресурсам Интернета.

Крупные предприятия, состоящие из нескольких территориально рассредоточенных отделений и филиалов, а также имеющие сотрудников, часто работающих дома, нуждаются в расширенном наборе услуг. Прежде всего, подобной услугой является такая транспортная услуга, как *виртуальная частная сеть* (Virtual Private Network, VPN), когда оператор связи создает для предприятия иллюзию того, что все его отделения и филиалы соединены частной сетью, то есть сетью, полностью принадлежащей предприятию-клиенту и полностью управляемой предприятием-клиентом. На самом же деле для создания этой иллюзии используется компьютерная сеть оператора, то есть общедоступная сеть, которая одновременно передает данные многих клиентов. В главе 22 мы рассмотрим эту важную услугу более подробно.

Корпоративные пользователи все чаще получают не только транспортные, но и информационные услуги операторов, например услуги хостинга, переносят собственные серверы, веб-сайты и базы данных на территорию оператора, поручая последнему поддерживать их работу и обеспечивать быстрый доступ к ним для сотрудников предприятия и, возможно, других пользователей сети оператора. Получившие в последнее время распространение облачные сервисы усилили эту тенденцию, позволяя корпоративным пользователям (и индивидуальным тоже) получать информационные услуги прозрачным способом, не заботясь об установке, конфигурировании и сопровождении серверов и программного обеспечения.

Инфраструктура

На формирование набора предлагаемых оператором услуг оказывает серьезное влияние материально-технический фактор. Так, для оказания услуг по аренде каналов оператор должен иметь в своем распоряжении первичную сеть SDH/OTN/DWDM, а для оказания услуг виртуальных частных сетей — маршрутизаторы с функциональностью MPLS или коммутаторы Carrier Ethernet.

Типичная сеть оператора связи имеет двухслойную структуру, с нижним уровнем первичной сети, служащей фундаментом для двух наложенных сетей — телефонной сети и компьютерной глобальной сети. Телефонная и глобальная компьютерная сети чаще всего представляют собой параллельные инфраструктуры, не связанные или слабо связанные друг с другом. Недостаточная интеграция этих сетей объясняется тем, что комбинированные услуги все еще не стали массовым продуктом операторов связи.

Структура глобальной компьютерной сети оператора связи в целом соответствует обобщенной структуре сети, описанной в разделе «Классификация компьютерных сетей» главы 4. Она состоит из магистральной сети, сетей агрегирования трафика и сетей доступа (рис. 18.2).

Коммуникационное оборудование пользователей взаимодействует с пограничным оборудованием сети доступа оператора связи по некоторому интерфейсу, называемому **интерфейсом пользователь—сеть** (User Network Interface, **UNI**).

Для предоставления информационных компьютерных услуг в сети имеется **центр данных**. Он представляет собой локальную сеть с серверами и соответствующим программным обеспечением веб-сервисов, IP-телефонии, облачных сервисов и т. п. Там могут также находиться серверы пользователей, если оператор предоставляет услуги *хостинга*. Отдельную группу ресурсов центра данных составляют серверы и программы систем управления сетью, помогающие администратору сети выполнять свою работу. Сети центров данных обычно присоединены непосредственно к магистрали сети (как показано на рис. 18.2),

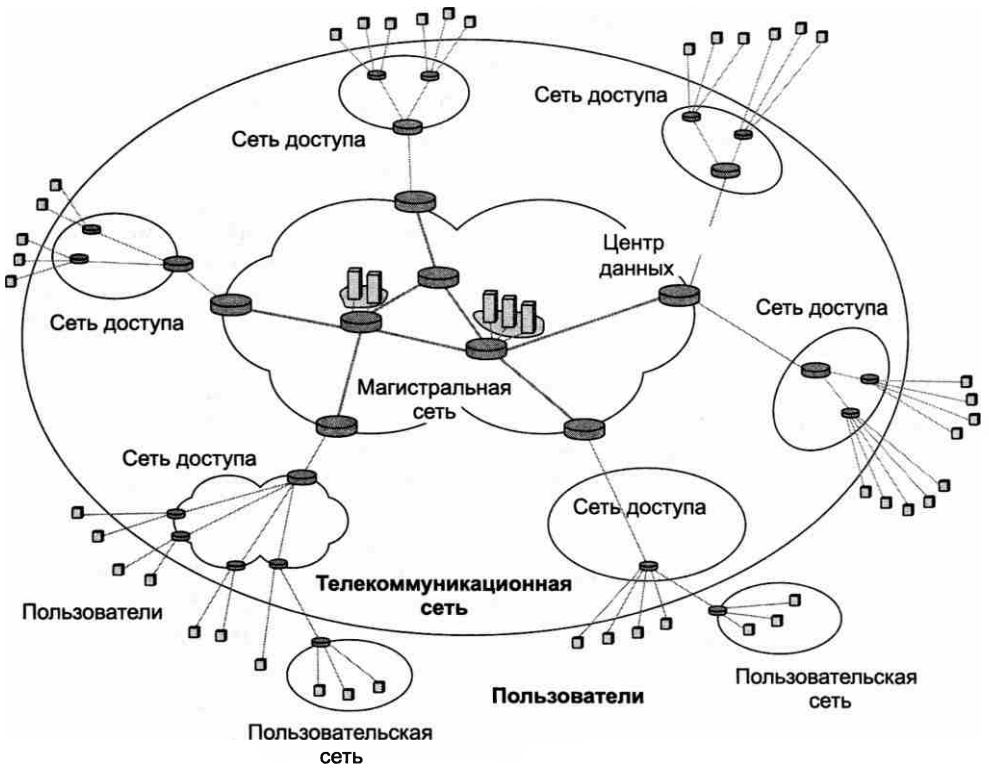


Рис. 18.2. Обобщенная структура глобальной компьютерной сети

чтобы обеспечить быстрый доступ к информационным ресурсам всем пользователям сети независимо от того, к какой именно сети доступа они подключены. Вместе с тем возможно также подключение центров данных к сетям агрегирования трафика для приближения их к конечным пользователям.

В тех случаях, когда у оператора отсутствует вся необходимая инфраструктура для оказания некоторой услуги, он может воспользоваться возможностями другого оператора; требуемая услуга может быть сконструирована на базе инфраструктуры партнера, а также собственных элементов инфраструктуры. Например, оператор связи может создать общедоступный веб-сайт электронной коммерции, не имея собственной IP-сети, соединенной с Интернетом. Для этого ему достаточно создать информационное наполнение сайта и разместить его на компьютере другого оператора, сеть которого имеет подключение к Интернету. Другим типичным примером такого рода является аренда оператором физических каналов связи для создания собственной телефонной или компьютерной сети, с тем чтобы на ее основе оказывать услуги своим клиентам. Оператора, который предоставляет услуги другим операторам связи, часто называют **оператором операторов** (carrier of carriers).

Каждая из сетей — магистральная, агрегирования трафика и доступа — предъявляет специфические требования к транспортным технологиям. Мы увидим, какие технологии преобладают в сетях каждого типа в следующей главе при рассмотрении стека протоколов глобальной сети.

Территория покрытия

По степени покрытия территории, на которой предоставляются услуги, операторы делятся на локальных, региональных, национальных и транснациональных.

Локальный оператор работает на территории города или сельского района. Традиционный локальный оператор владеет всей соответствующей транспортной инфраструктурой: физическими каналами между помещениями абонентов (квартирами, домами и офисами) и узлом связи, автоматическими телефонными станциями (АТС) и каналами связи между телефонными станциями. Сегодня к традиционным локальным операторам добавились альтернативные операторы, которые часто являются поставщиками услуг нового типа, прежде всего услуг Интернета, но иногда конкурируют с традиционными операторами и в секторе телефонии.

Региональные и **национальные операторы** оказывают услуги на большой территории, располагая соответствующей транспортной инфраструктурой. Традиционные операторы этого масштаба выполняют транзитную передачу телефонного трафика между телефонными станциями локальных операторов, имея в своем распоряжении крупные транзитные АТС, связанные высокоскоростными физическими каналами связи. Это — операторы операторов, их клиентами являются, как правило, локальные операторы или крупные предприятия, имеющие отделения и филиалы в различных городах региона или страны. Располагая развитой транспортной инфраструктурой, такие операторы обычно оказывают услуги дальней связи, передавая транзитом большие объемы информации без какой-либо обработки.

Транснациональные операторы оказывают услуги в нескольких странах. Они имеют собственные магистральные сети, покрывающие иногда несколько континентов. Часто подобные операторы тесно сотрудничают с национальными операторами, используя их сети доступа для доставки информации клиентам.

Взаимоотношения между операторами связи

Взаимосвязи между операторами различного типа (а также их сетями) иллюстрирует рис. 18.3. На рисунке показаны клиенты двух типов — индивидуальные и корпоративные. Нужно иметь в виду, что каждый клиент обычно нуждается в услугах двух видов — телефонных и передачи данных. Индивидуальные клиенты имеют в своих домах или квартирах, как правило, телефон и компьютер, а у корпоративных клиентов имеются соответствующие сети — телефонная, поддерживаемая офисным телефонным коммутатором (PBX), и локальная сеть передачи данных, построенная на собственных коммутаторах.

Для подключения оборудования клиентов операторы связи организуют так называемые **точки присутствия** (Point Of Presents, POP) — здания или помещения, в которых размещается оборудование доступа, способное подключить большое количество каналов связи, идущих от клиентов. Иногда такую точку называют **центральным офисом** (Central Office, CO) — это традиционное название для операторов телефонных сетей. К POP локальных операторов подключаются абоненты, а к POP операторов верхних уровней — операторы нижних уровней или крупные корпоративные клиенты, которым необходимы высокие скорости доступа и большая территория покрытия, объединяющая их офисы и отделения в разных городах и странах.

Так как процесс конвергенции пока еще не привел нас к появлению единой сети для всех видов трафика, то за каждым овалом, представляющим на этом рисунке сети операторов,

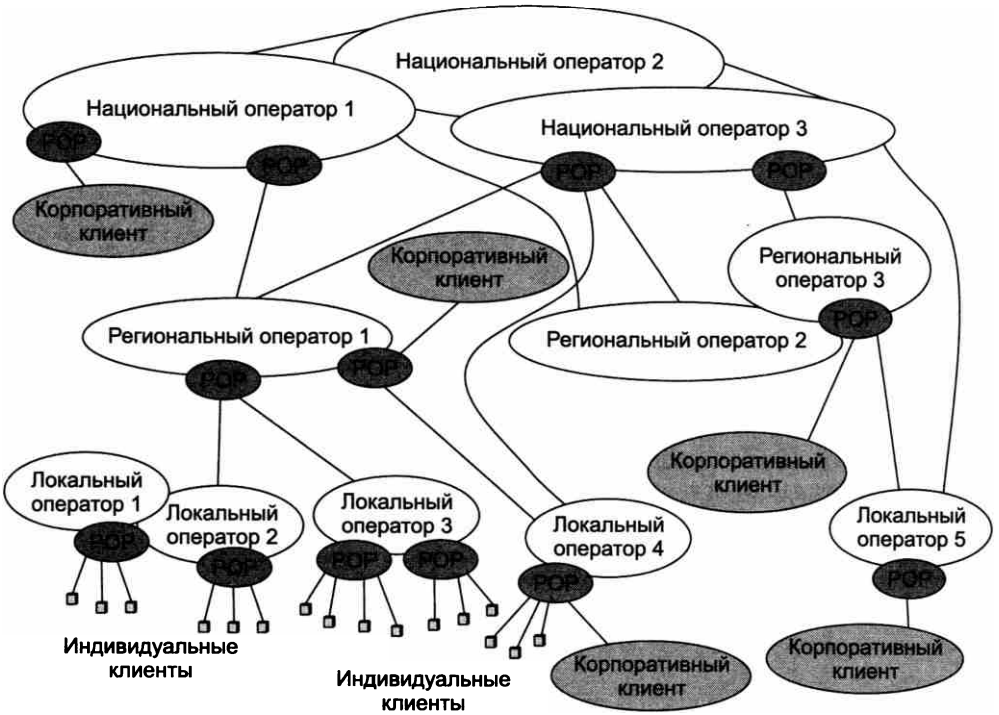


Рис. 18.3. Взаимоотношения между операторами связи различного типа

стоят две сети — телефонная и компьютерная (но опирающиеся на один и тот же фундамент — первичную сеть).

Как видно из рисунка, в современном конкурентном телекоммуникационном мире нет строгой иерархии операторов, взаимосвязи между ними и их сетями могут быть достаточно сложными и запутанными. Например, сеть локального оператора 5 имеет непосредственную связь не только с сетью регионального оператора 3, как того требует иерархия, но и непосредственную связь с национальным оператором 3 (возможно, этот оператор предлагает более дешевые услуги по передаче международного трафика, чем это делает региональный оператор 3). Некоторые операторы могут не иметь собственной транспортной инфраструктуры (на рисунке это локальный оператор 1). Как это часто бывает в таких случаях, локальный оператор 1 предоставляет только дополнительные информационные услуги, например предлагает клиентам локального оператора 2 видео по требованию или разработку и поддержание их домашних страниц в Интернете. Свое оборудование (например, видеосервер) такой оператор часто размещает в POP другого оператора, как это и показано в данном случае.

Организация Интернета

Интернет представляет собой уникальную глобальную компьютерную сеть, так как почти все существующие компьютерные сети (за исключением разве что некоторых сетей, требующих особых мер защиты от вторжений и поэтому полностью изолированных от

Интернета) и отдельные компьютеры являются частью этой сети. Поэтому протокол IP является обязательным и единственным протоколом сетевого уровня, объединяющим все сети в Интернете.

Уникальность Интернета проявляется во многих отношениях.

Прежде всего, это *самая большая в мире сеть*: по числу пользователей, по территории покрытия, по суммарному объему передаваемого трафика, по количеству входящих в ее состав сетей. Темпы роста Интернета, хотя и снизились по сравнению с периодом интернет-революции середины 90-х годов, остаются очень высокими и намного превышают темпы роста телефонных сетей.

Интернет — это *сеть, не имеющая единого центра управления* и в то же время работающая по единым правилам и предоставляющая всем своим пользователям единый набор услуг. Интернет — это «сеть сетей», но каждая входящая в Интернет сеть управляется независимым оператором — **провайдером услуг Интернета** (Internet Service Provider, ISP). Некоторые центральные органы существуют, но они отвечают только за единую *техническую политику*, за согласованный набор технических стандартов, за централизованное назначение таких жизненно важных для гигантской составной сети параметров, как имена и адреса компьютеров и входящих в Интернет сетей, но не за ежедневное поддержание сети в работоспособном состоянии. Такая высокая степень децентрализации имеет свои достоинства и недостатки.

Достоинства проявляются, например, в легкости наращивания Интернета. Так, новому поставщику услуг достаточно заключить соглашение по крайней мере с одним из существующих провайдеров, после чего пользователи нового провайдера получают доступ ко всем ресурсам Интернета. *Негативные* последствия децентрализации заключаются в сложности модернизации технологий и услуг Интернета. Любые коренные изменения требуют согласованных усилий всех провайдеров услуг, в случае «одного собственника» они проходили бы намного легче. Недаром многие новые технологии пока применяются только в пределах сети одного поставщика: примером может служить технология групповой рассылки, которая очень нужна для эффективной организации аудио- и видеовещания через Интернет, но все еще пока не может преодолеть границы, разделяющие сети различных провайдеров. Другой пример — не очень высокая надежность услуг Интернета, так как никто из провайдеров не отвечает за конечный результат, например за доступ клиента *A* к сайту *B*, если они находятся в сетях разных поставщиков.

Стремительный рост числа пользователей Интернета, привлекаемых информацией, содержащейся на его сайтах, изменил отношение корпоративных пользователей и операторов связи к этой сети. Сегодня Интернет поддерживается практически всеми традиционными операторами связи. Кроме того, к ним присоединилось большое количество новых операторов, построивших свой бизнес исключительно на услугах Интернета.

Поэтому общая структура Интернета, показанная на рис. 18.4, во многом является отражением общей структуры всемирной телекоммуникационной сети, фрагмент которой мы уже видели на рис. 18.3.

Магистральные провайдеры услуг являются аналогами транснациональных операторов связи. Они обладают собственными транспортными магистралями, покрывающими крупные регионы (страна, континент, весь земной шар). Примерами магистральных провайдеров услуг являются такие компании, как Cable & Wireless, WorldCom, Global One.

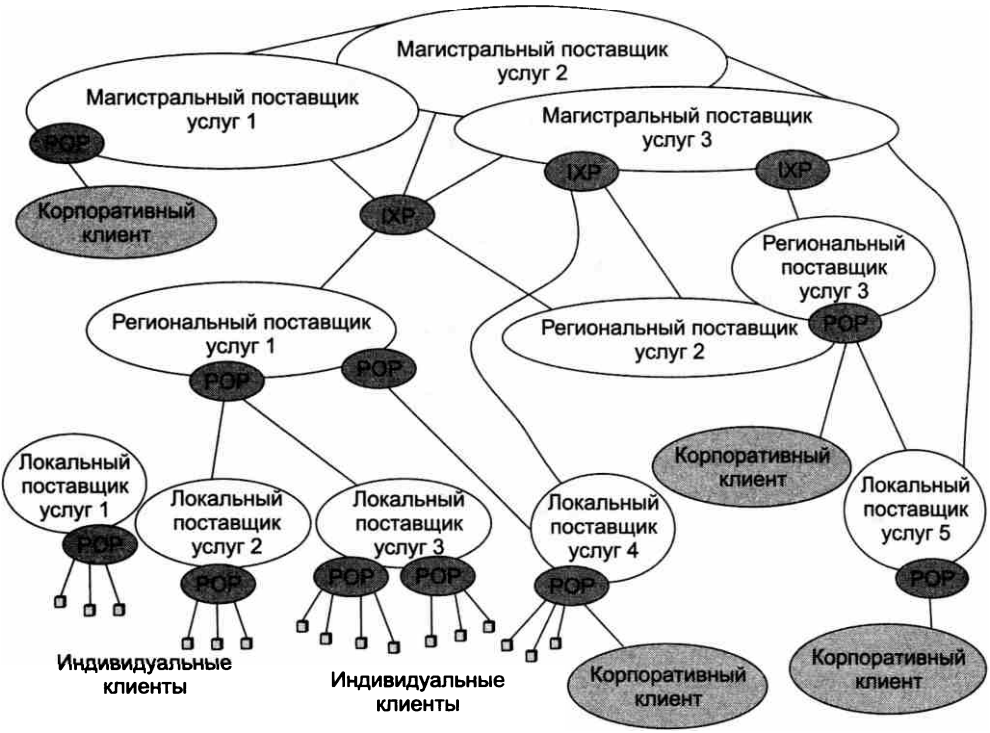


Рис. 18.4. Структура Интернета

Соответственно **региональные провайдеры услуг** оказывают услуги Интернета в рамках определенного региона (штат, графство, округ — в зависимости от принятого в той или иной стране административного деления), а **локальные провайдеры услуг** работают, как правило, в пределах одного города.

Связи между поставщиками услуг строятся на основе двусторонних соглашений о взаимной передаче трафика. Такие соглашения называют **пиринговыми** (от *англ.* peer — равный в статусе). Магистральный оператор обычно имеет пиринговые соглашения со всеми остальными магистральными операторами (так как их не много), а региональные операторы, как правило, заключают такие соглашения с одним из магистральных операторов и с несколькими другими региональными операторами.

Для того чтобы провайдерам было проще организовывать свои пиринговые связи, в Интернете существуют специальные **центры обмена трафиком**, в которых соединяются сети большого количества провайдеров. Такие центры обмена обычно называются Internet eXchange Point (IXP), или Network Access Point (NAP).

Центр обмена трафиком является средством реализации пиринговых связей, для этого он предоставляет поставщикам услуг помещение и стойки для установки коммутационного оборудования. Все физические и логические соединения между своим оборудованием провайдеры услуг выполняют самостоятельно. Это означает, что не все сети провайдеров, которые пользуются услугами того или иного центра обмена данными, автоматически обмениваются трафиком друг с другом, обмен происходит между сетями только в том слу-

чае, когда между провайдерами заключено пиринговое соглашение и они его реализовали в данном центре обмена.

В Интернете существует неофициальная градация *провайдеров Интернета* по уровням (tiers) в зависимости от того, кто из них и кому платит за передачу транзитного трафика Интернета. Провайдеры верхнего уровня (**Tier 1** — это, как правило, провайдеры международного и национального масштаба) могут достичь любой части Интернета без платы за транзитный трафик: у них у всех имеются некоммерческие пиринговые соглашения друг с другом. Провайдеры второго уровня (**Tier 2**) относятся к смешанному типу: с одними провайдерами у них имеются некоммерческие пиринговые соглашения, с другими — договоры о плате за транзит своего трафика. И наконец, провайдеры третьего уровня (**Tier 3**) совсем не имеют бесплатных пиринговых соглашений и платят другим провайдерам за транзит своего трафика.

Многослойное представление технологий и услуг глобальных сетей

Многоуровневый стек транспортных протоколов

Стек транспортных протоколов оператора связи состоит из нескольких уровней. Соответственно сеть оператора связи состоит из нескольких слоев оборудования, их число может быть меньше, чем число уровней реализуемого стека протоколов, так как некоторые коммуникационные устройства могут выполнять функции протоколов нескольких смежных уровней: например, мультиплексор DWDM может включать модули мультиплексора OTN.

Протокол определенного уровня может быть использован в двух целях:

- для предоставления услуг протоколам вышележащих уровней сети оператора;
- для реализации транспортных услуг клиентов.

Обобщенная структура слоев типичной сети оператора связи, который также играет роль поставщика услуг Интернета, показана на рис. 18.5.

На рисунке показаны уровни протоколов сетей с коммутацией пакетов, то есть компьютерных сетей, и слои технологий первичных сетей, которые используют принцип коммутации каналов.

Модель OSI не различает уровни протоколов сетей с коммутацией каналов, для нее они все представляют один физический уровень (вместе со средой передачи данных), но для понимания организации сети оператора связи полезно разбивать этот уровень как минимум на три слоя (что и сделано на рисунке):

- слой технологии DWDM оперирует с каналами-волнами (иногда его называют нулевым уровнем);
- слой технологии SDH/OTN оперирует цифровыми двухточечными каналами (первый уровень);
- слой физической среды строится на волоконно-оптических и медных кабелях, а также на беспроводной среде.

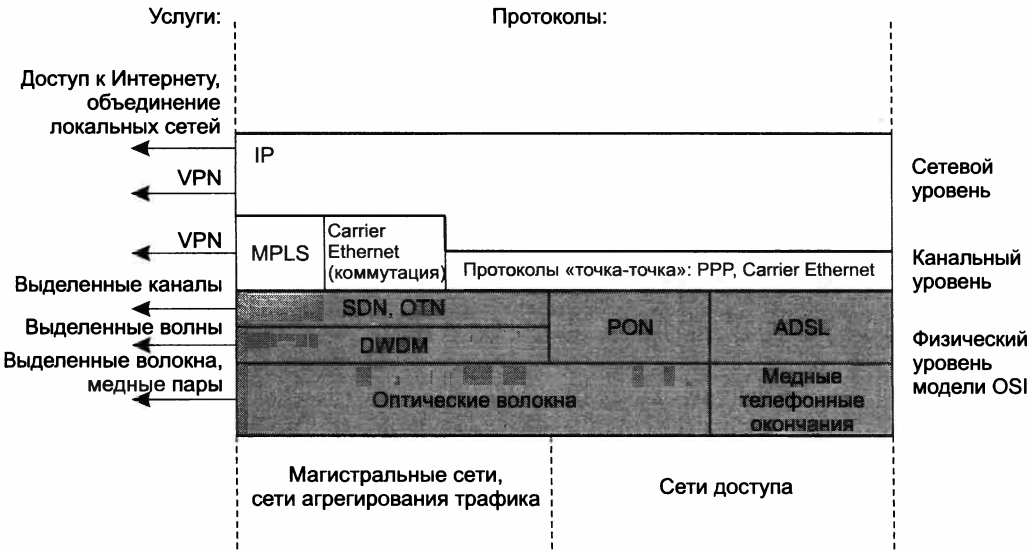


Рис. 18.5. Многослойная структура сети оператора связи/поставщика услуг Интернета

Внутренняя структура слоя технологий SDH и OTN, состоящая из нескольких уровней протоколов, на рисунке не отражена, она несущественна при укрупненном рассмотрении стека протоколов глобальной сети оператора связи.

Так как в этой части книги мы рассматриваем *транспортные* технологии глобальных сетей, то наш интерес заканчивается уровнем протокола IP, то есть сетевым уровнем, который является высшим обязательным уровнем протоколов транспортной подсистемы сети.

На рисунке показано, что на всех уровнях и слоях, кроме верхнего, существуют различные протоколы и технологии, решающие одни и те же задачи, но разными способами и с разной функциональностью. Например, на канальном уровне существуют протоколы MPLS, Carrier Ethernet, и PPP. Мы пока еще не познакомились с этими и другими протоколами, представленными новыми аббревиатурами на рисунке, но для понимания общей картины пока достаточно знать, что это протоколы канального уровня (можно только добавить, что Carrier Ethernet является версией Ethernet для сетей операторов связи, эта технология сохраняет все свойства коммутируемого варианта Ethernet, добавляя к ним некоторые дополнительные функции, полезные для мониторинга качества соединений в глобальной сети). Кроме того, одна и та же задача может решаться разными слоями: например, услуга VPN может предоставляться с помощью сетевого и канального уровней. Функциональность услуги в общем случае зависит от того, с помощью какого уровня она предоставляется.

Поэтому у оператора связи имеется возможность выбирать на каждом уровне из имеющегося набора однотипных протоколов какой-то один протокол, наиболее подходящий для решения его задач. Полученная в результате комбинация определяет специфический стек протоколов данной сети, например IP-MPLS-OTN-DWDM или IP-PPP-SDH-DWDM.

Необходимо также помнить и о территориальной структуре сети, то есть о том, что она состоит из магистральной сети, сетей агрегирования трафика и сетей доступа. Для магистральной сети и сетей доступа применяются практически одни и те же технологии,

отличия заключаются только в скорости каналов и протоколов: если в магистральной сети преобладают скорости 10 и 100 Гбит/с, то в сетях агрегирования трафика они на порядок ниже: 1 и 10 Гбит/с, что позволяет сбалансировать нагрузку этих сетей. В сетях доступа обычно применяются технологии, учитывающие специфику топологии (звезда, от офиса оператора связи до домов индивидуальных пользователей и зданий организаций) и линий связи (телефонные медные окончания, телевизионный кабель). Эта специфика отражена на рисунке.

Технологии и услуги физического уровня

Особенностью глобальных сетей является сложная структура физического уровня. На физическом уровне локальных сетей используются только кабели. В глобальных же сетях для создания канала между двумя коммутаторами или маршрутизаторами, как правило, применяются устройства первичных сетей, такие как мультиплексоры или кросс-коннекторы сетей PDH, SDH, OTN или DWDM, о которых мы достаточно подробно писали в главе 10.

Первоначально технологии первичных сетей предназначались только для внутренних целей операторов связи в качестве гибкого средства соединения телефонных коммутаторов, то есть для гибкого создания каналов между их собственными коммутаторами, изначально телефонными, а потом и пакетными. Постепенно, с ростом популярности компьютерных сетей технологии первичных сетей стали применяться для предоставления транспортных услуг конечным пользователям.

На рис. 18.5 показаны три типа услуг, которые предоставляются операторами связи с помощью трех нижних слоев их сети:

- *Услуга выделенных оптических волокон.* Обычно эту услугу один оператор, обладающий развитой кабельной инфраструктурой со свободными оптическими кабелями или волокнами, оказывает другому оператору, который затем строит на этих волокнах собственную первичную сеть, соединяя с помощью волокон мультиплексоры DWDM/OTN или SDH. Волокна, сдаваемые в аренду, часто называют **темными волокнами** (dark fibre), так как они не подключены к оборудованию передачи данных и не «подсвечены» лазерными передатчиками.
- *Услуга выделенных волновых каналов.* Потребителями этой услуги могут быть как операторы связи, так и корпоративные пользователи. Обычно такая услуга предоставляется в формате кадров OTN или SDH высшего уровня иерархии скорости, который в настоящее время для обеих технологий равен 100 Гбит/с. Пользователь может задействовать волновой канал для построения собственной первичной сети, соединяя таким образом свои мультиплексоры OTN или SDH, а может непосредственно соединить IP-маршрутизаторы, имеющие соответствующие интерфейсы (OTN или SDH). Обычно IP-маршрутизаторы обладают так называемыми серыми интерфейсами SDH или OTN; это означает, что они работают с неокрашенными волнами, соответствующими центру окна прозрачности, например с волной 1310 нм. Для того чтобы использовать определенную волну DWDM, которая отличается от «серой» волны, например волну 1528,77 нм, необходим *транспондер* — устройство преобразования длин волн. Транспондер является частью мультиплексора DWDM, принимающего «серую» волну от маршрутизатора и преобразующего ее в волну нужного цвета для дальнейшей передачи по сети DWDM. Новой тенденцией являются маршрутизаторы с «окрашенными»

интерфейсами, то есть с интерфейсами, которые смогут настраиваться на генерацию определенной волны из сетки частот DWDM, в этом случае транспондеры мультиплексоров DWDM не нужны.

- *Услуга выделенного соединения по протоколу OTN, SDH или PDH.* Это наиболее традиционная услуга оператора связи; она была очень востребована в 1980–1990 годы корпоративными клиентами, которые, соединяя выделенными каналами свои IP-маршуртизаторы, строили собственную корпоративную компьютерную сеть. Со временем услуги выделенных каналов для построения корпоративной сети стали вытесняться более дешевыми и гибкими услугами глобальных сетей с коммутацией пакетов (frame relay, ATM, MPLS) и Интернета. Операторы IP-сетей, не имеющие своей инфраструктуры физических каналов связи, по-прежнему пользуются этими услугами, покупая их у операторов связи.

Технологии и услуги сетей коммутации пакетов

Примером услуг, предоставляемых операторами связи на основе технологий канального и сетевого уровней, являются доступ в Интернет и связывание территориально разнесенных локальных сетей.

Доступ в Интернет — это услуга, предоставляемая операторами связи — провайдерами Интернета. Благодаря тому, что IP-сети операторов связи объединены в глобальную сеть Интернет, потребитель этой услуги теоретически получает доступ ко *всем* узлам Интернета и ко *всем* их услугам. В соответствии с принципами работы IP-сетей даже в том случае, когда интересующий клиента узел находится в сети, не принадлежащей провайдеру услуги, пакеты клиента могут его достичь через сети других операторов. Услуга доступа в Интернет является *транспортной*, то есть сама по себе она не предоставляет никаких прикладных сервисов, таких как веб-сервис или сервис IP-телефонии. Эти прикладные сервисы работают поверх службы доступа в Интернет и для самого транспорта Интернета они прозрачны.

Объединение локальных сетей клиентов может выполняться как на IP-уровне, так и на канальном уровне. Определяющим в выборе решения, на каком уровне объединять сети, является тип адресации, используемый для этой операции. Если сети объединяются на основе их IP-адресов, то это объединение на IP-уровне. Если же это объединение происходит без учета IP-адресов узлов объединяемой сети, а с учетом адресов канального уровня, то это объединение на канальном уровне. Обратите внимание, что в том и в другом случае объединяемые сети обмениваются IP-пакетами, которые инкапсулированы в кадры канального уровня, однако при оказании услуги канального уровня эти пакеты не принимаются во внимание.

На IP-уровне объединение локальных сетей может быть организовано как дополнительная услуга на основе услуги доступа в Интернет. Для этого оператор должен выделить клиенту пул публичных IP-адресов для назначения их узлам локальных сетей и обеспечить маршрутизацию этих адресов в Интернете.

Услуга виртуальных частных сетей (VPN) является важным типом услуги объединения локальных сетей, так как она обладает несколькими критичными для клиентов свойствами, создающими эффект изолированности клиентских сетей. Она может предоставляться как на IP-уровне, так и на канальном уровне.

При объединении сетей клиентов на канальном уровне эти сети обмениваются трафиком через сеть канального уровня провайдера услуги, то есть через сеть MPLS или Carrier Ethernet. Маршрутизаторы провайдера услуг в этой операции не участвуют, трафик клиентских сетей в них не заходит.

Модели межуровневого взаимодействия в стеке протоколов глобальной сети

Как мы знаем, каждый уровень стека протоколов, кроме верхнего, оказывает внутренние транспортные услуги следующему уровню протоколов иерархии стека, перенося его сообщения.

Имеется принципиальное различие в функциональности такой транспортной услуги в зависимости от того, выполняет ли данный уровень коммутацию своих кадров (другими словами, есть ли на данном уровне сеть коммутаторов) или же он служит только для «кадрирования» сообщений верхнего уровня, а коммутация выполняется на других уровнях стека протоколов. От того, как реализован тот или иной уровень стека протоколов — по первому варианту (коммутация) или по второму (кадрирование), — зависит функциональность многослойной глобальной сети и ее возможности по предоставлению транспортных услуг: понятно, что уровень, в котором не поддерживается коммутация, не может использоваться для предоставления независимых услуг, у него просто отсутствуют для этого возможности.

В многоуровневой модели стека протоколов глобальной сети есть два уровня, которые могут быть реализованы по первому или второму варианту — это канальный уровень сети с коммутацией пакетов и верхний слой первичной сети, то есть слой OTN или SDH.

Поддержка IP-маршрутизаторов канальным уровнем. В первом варианте на канальном уровне работает сеть, выполняющая коммутацию кадров (рис. 18.6). Сегодня это может быть сеть MPLS или Ethernet.

В этом варианте IP-маршрутизаторы не имеют непосредственных физических связей между собой (то есть связей любого типа, обеспечиваемых физическим уровнем, — каналов SDH, OTN, DWDM или кабельных связей). Вместо этого они соединены такими связями с коммутаторами канального уровня, например коммутаторами Carrier Ethernet. Сеть коммутаторов канального уровня обеспечивает передачу пакетов между IP-маршутизаторами. Например, если маршрутизатору R1 нужно передать пакет маршрутизатору R2, то он отправляет его коммутатору S1 с указанием адреса канального уровня, ведущего через сеть этого уровня к интерфейсу маршрутизатора R2. Сеть канального уровня сама решает задачу маршрутизации пакета, в нашем примере этот маршрут проходит через промежуточный коммутатор S2, а затем коммутатор S3 передает пакет маршрутизатору назначения, то есть R2.

Сеть канального уровня в этом варианте может использоваться и для предоставления услуг на своем уровне для внешних потребителей.

Во втором варианте у оператора нет сети канального уровня, а IP-маршрутизаторы непосредственно связаны друг с другом с помощью связей физического уровня (рис. 18.7).

Отсутствие сети канального уровня не означает, что протокол канального уровня отсутствует в стеке протоколов сети оператора связи. Просто он представлен только в интерфейсах маршрутизаторов, которые инкапсулируют в его кадры IP-пакеты и отправляют

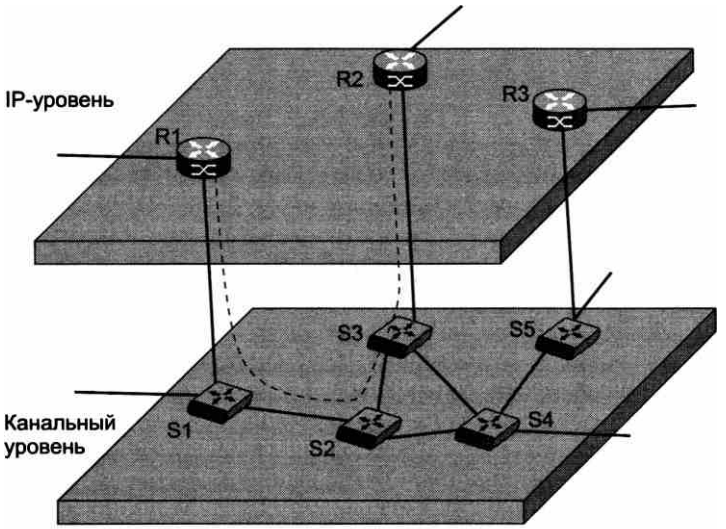


Рис. 18.6. Взаимодействие IP-сети с сетью канального уровня

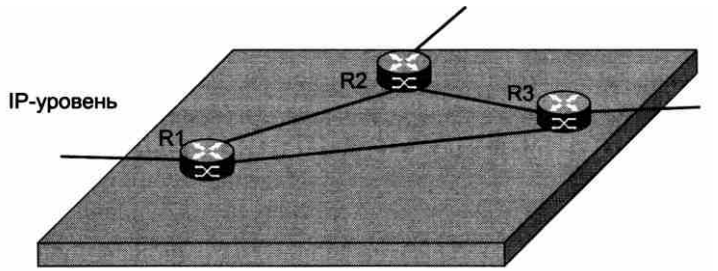


Рис. 18.7. Непосредственное взаимодействие IP-маршрутизаторов

их по физическому каналу связи интерфейсу следующего маршрутизатора, а техника коммутации пакетов в этом варианте работает только на IP-уровне. Канальный уровень используется лишь для оформления кадров, например кадров Ethernet, то есть выполняет функции кадрирования IP-пакетов. Но и в этом урезанном качестве протокол канального уровня может быть полезен, если он выполняет функции, не поддерживаемые протоколом IP, например обеспечивает контроль достоверности получаемых кадров за счет контрольной суммы или позволяет выполнять мониторинг временны́х характеристик потока пакетов. Однако независимые транспортные услуги, такие как услуги VPN канального уровня, с помощью этого варианта канального уровня предоставлять нельзя. Иногда такую модель называют «чистой» IP-сетью, имея в виду тот факт, что коммутация (маршрутизация) выполняется только на IP-уровне.

На рис. 18.5 существование двух вариантов работы IP-протокола с канальным уровнем отражается высотой канального уровня: он выше в тех случаях, когда на канальном уровне имеется сеть с коммутацией кадров (MPLS, Carrier Ethernet), и ниже для случаев ее отсутствия (PPP, Carrier Ethernet). Протокол Carrier Ethernet попал в оба варианта, так

как с его помощью можно как построить сеть коммутаторов Ethernet, так и применять этот протокол только на интерфейсах IP-маршрутизаторов. Протокол PPP разработан специально для работы в двухточечной топологии и коммутацию кадров он не поддерживает.

У каждого из двух рассмотренных вариантов имеются свои достоинства и недостатки. Кроме уже обсужденного преимущества канального уровня с коммутацией по предоставлению внешних услуг, этот вариант может также помочь разгрузить IP-маршрутизаторы, передавая часть устойчивых и высокоскоростных потоков данных исключительно средствами канального уровня. С другой стороны, вариант без сети канального уровня проще, так как на один уровень оборудования в сети становится меньше.

Поддержка IP-маршрутизаторов оптическими технологиями. В том случае, когда IP-маршрутизаторы соединены непосредственно каналами физического уровня (то есть коммутация на канальном уровне отсутствует), могут работать две популярные модели стека протоколов, называемые **IP поверх OTN** и **IP поверх DWDM**. Они отличаются наличием коммутации в слое OTN глобальной сети. Модель «IP поверх SDH» также применяется, для рассматриваемого нами вопроса она аналогична модели «IP поверх OTN».

В модели «IP поверх OTN» IP-маршрутизаторы соединяются с помощью каналов, образованных OTN-коммутаторами. OTN/SDH-коммутация позволяет создавать каналы различной пропускной способности, от 2,5 до 100 Гбит/с (в случае OTN-коммутации), что дает оператору возможность строить магистральную IP-сеть и IP-сети агрегирования трафика достаточно гибко — из маршрутизаторов с различной производительностью и скоростью интерфейсов.

В модели «IP поверх DWDM» отсутствует уровень OTN-коммутации, IP-маршрутизаторы подключаются непосредственно к портам DWDM-мультиплексора. В этой модели OTN присутствует только как «кадрирующая» технология, то есть IP-пакеты инкапсулируются в OTN-кадры, которые принимаются портами мультиплексора DWDM, также использующими OTN-кадрирование. Коммутация на физическом уровне происходит только на основе техники DWDM, то есть коммутируются волны, но не блоки данных OTN. Учитывая, что коммутация волн пока не обладает большой гибкостью, можно считать, что IP-маршрутизаторы соединены в этой модели постоянными связями, а коммутация происходит только на IP-уровне.

Для обеспечения высокой производительности магистрали интерфейсы IP-маршрутизаторов в этой модели обычно поддерживают максимально возможную скорость передачи данных, предоставляемую мультиплексорами DWDM, которой сегодня является скорость 100 Гбит/с. Модель «IP поверх DWDM» применяется для построения магистралей сетей. Для сетей агрегирования трафика она недостаточно гибкая (только один уровень скорости интерфейсов) и слишком дорогая (интерфейсы маршрутизаторов, способные работать с мультиплексорами DWDM, должны быть «окрашенными», то есть генерировать волну из частотного плана DWDM, а это требует установки в интерфейсе дорогостоящих оптических компонентов).

Мы рассмотрели два типа моделей стека глобальной сети — «IP поверх DWDM» и «IP поверх OTN», отличающихся наличием или отсутствием коммутации в слое OTN/SDH, при этом обе модели подразумевают отсутствие коммутации на канальном уровне. Применяются и две другие модели, в которых имеется коммутация на канальном уровне. В том случае, когда коммутация имеется и на канальном уровне, и на уровне OTN, говорят о пол-

ной модели стека. Четвертая модель, с коммутацией на канальном уровне и ее отсутствием на уровне OTN, специального названия не имеет.

Выводы

Глобальная компьютерная сеть является, как правило, частью глобальной телекоммуникационной сети оператора связи.

Классификация глобальных сетей может быть выполнена на основе различных критериев. Это могут быть технологические характеристики сетей, такие как топология, метод коммутации, метод продвижения пакетов, тип используемой среды передачи. Сети классифицируют и на основе других признаков, не являющихся технологическими, таких, например, как тип клиентов, набор услуг, территория покрытия, место в иерархии провайдеров услуг.

Глобальные сети предоставляют услуги двух типов: информационные и транспортные.

Основными типами транспортных услуг глобальных компьютерных сетей являются услуги выделенных линий, доступа в Интернет и виртуальных частных сетей (VPN).

Большинство современных глобальных сетей являются составными IP-сетями, а отличия между ними заключаются в технологиях, лежащих под уровнем IP.

Крупные глобальные сети обычно строятся по многослойной схеме, где два нижних слоя — это слои первичной сети, образуемые технологиями DWDM и OTN/SDH. На основе первичной сети оператор сети строит каналы наложенной (оверлейной) сети — пакетной или телефонной. IP-сеть образует верхний уровень.

Каждый слой такой сети может выполнять две функции:

- предоставление услуг конечным пользователям;
- поддержка функций вышележащих уровней сети оператора.

Соотношение понятий «технология» и «услуга» можно пояснить следующими утверждениями:

- одна и та же технология может быть использована для предоставления различных услуг: например, технология IP может быть использована как для доступа в Интернет, так и для организации виртуальной частной сети;
- одна и та же услуга может быть реализована на основе разных технологий: например, виртуальную частную сеть можно построить на основе технологии IP, Ethernet и MPLS;
- имеются такие услуги, которые можно предоставлять на основе только какой-то одной специфической технологии: например, услугу выделенной волны можно предоставлять только на основе технологии DWDM.

В зависимости от того, выполняется ли коммутация на канальном уровне и в слое OTN/SDH, существует четыре модели организации стека протоколов глобальной сети:

- полная модель — коммутация выполняется как на канальном уровне, так и на уровне OTN/SDH;
- IP поверх DWDM — коммутация не выполняется ни на канальном уровне, ни на уровне OTN/SDH;
- IP поверх OTN — коммутация не выполняется на канальном уровне, но выполняется на уровне OTN/SDH;

- промежуточная модель — коммутация выполняется на канальном уровне, но не выполняется на уровне OTN/SDH.

Контрольные вопросы

1. К какому типу сети оператора связи обычно подключаются центры данных? Варианты ответов:
   а) к сети доступа;
   б) к сети агрегирования трафика;
   в) к магистральной сети.
2. Какие из перечисленных услуг относятся к транспортным:
   а) доступ в Интернет;
   б) передача файлов;
   в) сервис выделенных каналов.
3. Что из перечисленного является характеристикой Интернета:
   а) самая большая в мире сеть;
   б) сеть с коммутацией каналов;
   в) сеть сетей;
   г) сеть, работающая на протоколе IPX.
4. Верно ли утверждение, что в модели IP поверх DWDM отсутствует канальный уровень?
5. Услуга виртуальных частных сетей может предоставляться:
   а) на уровне IP;
   б) на канальном уровне;
   в) на физическом уровне.

ГЛАВА 19 Транспортные технологии глобальных сетей

Технологии виртуальных каналов — от X.25 к MPLS

Сегодня все глобальные компьютерные сети объединяют Интернет и протокол IP. В глобальных сетях протокол IP работает поверх специфических технологий канального уровня, разработанных с учетом характеристик глобальных линий связи и транспортных услуг, предоставляемых этим видом сетей.

На протяжении всего времени существования глобальных компьютерных сетей важную роль в них играли транспортные технологии, основанные на технике виртуальных каналов. Мы рассмотрим, каким образом происходила эволюция технологий этого типа, от X.25 через frame relay и ATM к MPLS — технологии, которая смогла объединить технику виртуальных каналов с протоколами управляющего слоя стека TCP/IP.

Принципы работы виртуального канала

В этом разделе мы напомним[1] о принципах работы виртуального канала и дополним ваши знания.

В сети с виртуальными каналами два узла могут начать обмен данными только после того, как между ними будет установлено логическое соединение — **виртуальный канал**. Виртуальный канал лучше защищает пользователей от внешних атак, поскольку у злоумышленника нет возможности посылать пакеты данных от одного произвольного узла к другому, что вполне можно сделать в сети, построенной на транспортной технологии дейтаграммного типа, такой как IP или Ethernet.

Продвижение кадров вдоль виртуального канала происходит не на основе адресов конечных узлов, а на основе *метки*, которая позволяет коммутаторам сети определять принадлежность кадров тому или иному виртуальному каналу и продвигать их соответственно. Значение метки потока изменяется в каждом коммутаторе при передаче кадра с входного интерфейса на выходной, — в этом случае говорят, что происходит коммутация по меткам. Коммутация по меткам позволяет избавиться от требования уникальности их значений в пределах сети, которую обеспечить сложно; для того, чтобы кадры различных виртуальных каналов не смешивались, достаточно обеспечить уникальность значений меток только в пределах отдельного интерфейса. Из-за того, что коммутация по меткам является

[1] См. главу 3.

неотъемлемым атрибутом технологий виртуальных каналов, у них есть и второе название — **технологии коммутации по меткам**.

Технически установление виртуального канала означает формирование записей в *таблицах продвижения кадров* на каждом коммутаторе вдоль виртуального канала. Такая таблица включает информацию о продвижении: на какой выходной порт нужно передать кадр с данной меткой, какое новое значение нужно присвоить метке после передачи кадра на выходной интерфейс.

На рис. 19.1 показан фрагмент сети, состоящей из двух коммутаторов S1 и S2 и четырех конечных узлов C1-C4. Через эти коммутаторы проложено три виртуальных канала: C1-C2, C1-C4 и C3-C4.

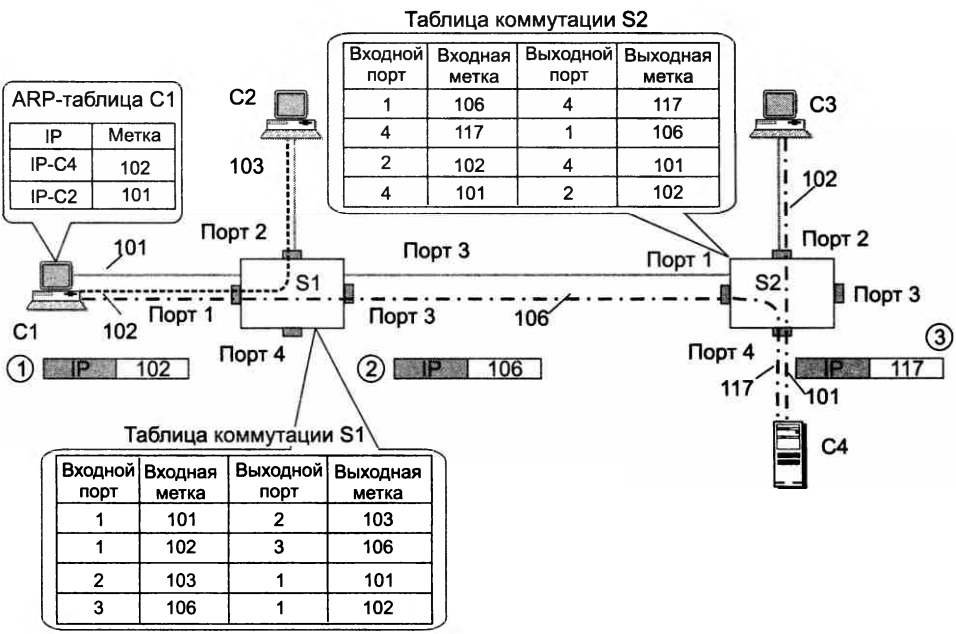


Рис. 19.1. Продвижение кадров вдоль виртуальных каналов

Эти каналы являются *двунаправленными*, а это означает, что кадры по ним могут передаваться в любом из двух направлений. Для каждого виртуального канала в таблице продвижения имеется две записи — по одной для каждого направления. Например, первая запись в таблице коммутации коммутатора S1 (запись 1-101-2-103) определяет работу коммутатора по продвижению кадров виртуального канала C1-C2 в направлении от C1 к C2; она предписывает коммутатору S1 передать кадр, принятый на порт 1 со значением метки 101, на порт 2 и поменять значение метки (скоммутировать метку) на 103. Третья запись (2-103-1-101) означает, что все пакеты, которые поступят на порт 2 со значением метки 102, будут продвигаться на порт 3, а ее значение поменяется на 101.

Существуют также *однонаправленные* виртуальные каналы. В случае их использования для дуплексного обмена информацией нужно установить два независимых виртуальных канала между конечными узлами, по одному для каждого направления.

Виртуальные каналы делятся на два класса:

- **коммутируемые виртуальные каналы** (Switched Virtual Circuit, **SVC**);
- **постоянные виртуальные каналы** (Permanent Virtual Circuit, **PVC**).

Создание коммутируемого виртуального канала происходит по инициативе конечного узла сети с помощью специального протокола, посылающего пакет с запросом на установление соединения в направлении к узлу назначения виртуального канала. Название «коммутируемый» отражает тот факт, что канал создается динамически по требованию узла-отправителя аналогично установлению коммутируемого соединения в телефонной сети. Для поддержания режима SVC в сети должны существовать таблицы маршрутизации, в соответствии с которыми продвигается пакет с запросом соединения. По отношению к *пакету с запросом соединения* сеть работает в дейтаграммном режиме, и такой пакет должен содержать *адрес назначения конечного узла, а не метку*.

Постоянный виртуальный канал устанавливается вручную, администратор создает его на достаточно длительное время (отсюда название), возможно, с привлечением централизованной системы управления сетью. Пограничный коммутатор сети принимает пакеты от внешней сети, которая может и не поддерживать технику виртуальных каналов. Пограничный коммутатор должен каким-то образом отображать приходящие извне пакеты на один из виртуальных каналов сети. В простейшем случае такое отображение (mapping) выполняется на основе входного физического интерфейса, то есть все кадры, приходящие на некоторый входной интерфейс, отображаются на один и тот же виртуальный канал. В более сложных случаях необходимо различать несколько потоков, приходящих на входной интерфейс, и отображать их на разные виртуальные каналы. В таком случае в пограничном коммутаторе наряду с таблицей продвижения должна существовать таблица отображения потоков. В примере рис. 19.1 такая таблица имеется у конечного узла C1. В ней в качестве признака потока используются IP-адреса назначения, поэтому таблица отображения представляет собой ARP-таблицу.

Виртуальные каналы чаще всего имеют *двухточечную топологию*. Однако существуют каналы и с другим типом топологии — *звезда* (рис. 19.2). В канале с такой топологией один и тот же кадр передается от источника — центра звезды, называемого также концентратором, — вдоль ее лучей всем конечным узлам. Конечные узлы не могут использовать виртуальный канал звездообразной топологии для обмена кадрами между собой, он передает кадры в обратном направлении только от конечного узла к центральному узлу. Виртуальные каналы со звездообразной топологией рассчитаны на эффективную поддержку группового вещания.

Виртуальный канал является удобным инструментом для *инжиниринга трафика*.

Это объясняется тем, что он может быть установлен независимо в каждом промежуточном коммутаторе путем соответствующего назначения локальных меток, выполняемого администратором сети или внешней программной системой. Поток пакетов, который должен быть передан по виртуальному каналу, может быть определен гибко и с любой степенью детализации, в этом определении могут использоваться не только IP-адреса назначения, как это происходит в IP-сетях, но и любые признаки: IP-адреса источника, TCP/UDP-порты назначения, поле DSCP и т. п., что также повышает эффективность инжиниринга трафика.

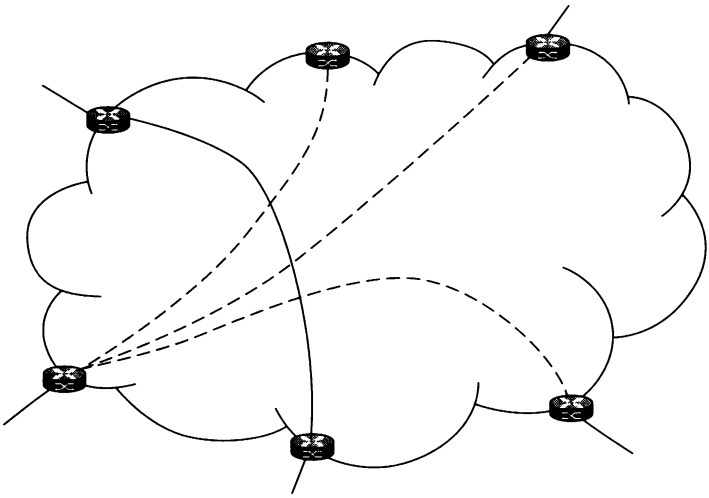


Рис. 19.2. Топологии виртуальных каналов

Сети, работающие на основе техники виртуальных каналов, относятся к типу **сетей, не поддерживающих широковещание с множественным доступом** (Non Broadcast Multiple Access, **NBMA**). Действительно, у такой сети существует произвольное количество конечных узлов, но отсутствует возможность послать кадр сразу всем узлам — ни двухточечный, ни звездообразный виртуальный канал этого не позволяет. В сетях NBMA протокол IP не может воспользоваться услугами протокола ARP для автоматического построения ARP-таблицы, так как эти услуги основаны на широковещательных запросах. Так что в тех случаях, когда входящий поток отображается на виртуальный канал на основе IP-адреса, таблицу отображения, которая является здесь ARP-таблицей, приходится строить вручную или же с помощью некоторого дополнительного протокола, не использующего широковещание.

Эффективность виртуальных каналов

Сравнение эффективности виртуальных каналов мы проведем отдельно для коммутируемых и постоянных виртуальных каналов, сравнивая первые с дейтаграммными технологиями, а вторые — с выделенными физическими каналами.

Применение *коммутируемых виртуальных каналов* требует предварительного установления соединения, что вносит дополнительную задержку перед передачей данных по сравнению с применением дейтаграммных протоколов. Эта задержка особенно сказывается при передаче небольшого объема данных, когда время установления виртуального канала может быть соизмеримым со временем передачи данных. Кроме того, дейтаграммный метод быстрее адаптируется к изменениям в сети. При отказе коммутатора или линии связи вдоль виртуального канала соединение разрывается, и виртуальный канал нужно прокладывать заново, обходя отказавшие участки сети.

Однако сравнивая эти два принципиально различных подхода, следует учесть, что время, затраченное на установление виртуального канала, компенсируется последующей быстрой передачей всего потока пакетов. Маршрутизация пакетов в сети с поддержкой виртуальных каналов ускоряется за счет двух факторов. Первый состоит в том, что решение о продвижении пакета принимается быстрее, так как таблица коммутации, в которой есть информация только об установленных виртуальных каналах, чаще всего существенно меньше таблицы маршрутизации, в которой число записей определяется количеством сетей назначения (размер таблицы маршрутизации магистральных IP-маршрутизаторов провайдеров Интернета составлял весной 2015 года около 550 000 записей).

Вторым фактором является уменьшение доли служебной информации в пакетах. Адреса конечных узлов в глобальных сетях обычно имеют достаточно большую длину — 4 байта в версии IPv4, 16 байт в версии IPv6, MAC-адрес имеет длину 6 байт. Номер же виртуального канала обычно занимает 10–12 бит, так что накладные расходы на адресную часть существенно сокращаются, а значит, полезная скорость передачи данных возрастает.

Постоянные виртуальные каналы являются гораздо более эффективными в отношении производительности передачи данных, чем коммутируемые. Значительную часть работы по маршрутизации пакетов сети выполняет администратор, вручную прокладывая постоянные виртуальные каналы и оставляя коммутаторам только продвижение пакетов на основе готовых таблиц коммутации портов.

Постоянный виртуальный канал подобен выделенному физическому каналу в том смысле, что для каждой операции обмена данными не требуется заново устанавливать или разрывать соединение. Отличие же состоит в том, что пользователь PVC не имеет гарантий относительно действительной пропускной способности канала. Зато применение PVC обычно намного дешевле, чем аренда выделенной линии, так как пользователь делит пропускную способность сети с другими пользователями.

Постоянные виртуальные каналы выгодно использовать для передачи *агрегированных потоков трафика*, состоящих из большого количества индивидуальных потоков абонентов сети. В этом случае виртуальный канал прокладывается не между конечными абонентами, а между участком магистрали сети, на котором данный агрегированный поток существует, например от одного пограничного маршрутизатора сети оператора связи до другого. В силу закона больших чисел агрегированные потоки обладают высокой степенью устойчивости, так что для них нет смысла динамически создавать коммутируемые виртуальные каналы — лучше эффективно использовать постоянные, которые при хорошем планировании (методами инжиниринга трафика) оказываются достаточно загруженными.

Подводя итог, можно сказать, что виртуальные каналы более эффективны при передаче долговременных, чем кратковременных, потоков, так как в этом случае снижаются удельные затраты на установление соединений.

Технология X.25

Технология виртуальных каналов X.25 появилась на заре эры компьютерных сетей, практически одновременно с сетью ARPANET, давшей начало Интернету и дейтаграммному протоколу IP. Долгое время, до середины 1980-х, X.25 была основной технологией для построения как сетей операторов связи, так и корпоративных сетей.

Технология X.25 оказалась хорошо приспособленной для построения глобальной всемирной сети благодаря тому, что была масштабируемой — в ней был определен протокол межсетевого взаимодействия, позволяющий объединять сети разных провайдеров, а также поддерживалась международная система иерархической адресации X.121, включающая код страны, номер сети и номер терминала в сети.

Сети X.25 используют трехуровневый стек протоколов. Физический уровень в то время чаще всего был представлен модемами, работающими на коммутируемых и выделенных телефонных линиях со скоростями 2400–9600 Кбит/с. Как на канальном (LAP-B), так и на сетевом (X.25/3) уровне протоколы стека X.25 поддерживают установление соединений и коррекцию ошибок на основе метода *скользящего окна*. Такая избыточность функций, направленных на обеспечение надежности передачи данных, объясняется ориентацией технологии на ненадежные аналоговые каналы. Распространение высокоскоростных и надежных цифровых оптических каналов в середине 80-х годов привело к тому, что функции технологии X.25 по обеспечению надежной передачи данных превратились из достоинства технологии в ее *недостаток*, так как лишь замедляли скорость передачи пользовательских данных. Результатом этой революции стало появление принципиально новой технологии глобальных сетей, а именно Frame Relay.

Технология Frame Relay

Главным достоинством Frame Relay является *простота*; освободившись от многих ненужных в условиях существования надежных оптических каналов связи функций, эта технология предлагает только тот минимум услуг, который необходим для быстрой доставки кадров адресату. В соответствии с этой концепцией протокол Frame Relay работает в режиме передачи данных по «возможности», то есть не поддерживает процедуры надежной передачи кадров, оставляя повторную передачу искаженных и потерянных данных протоколам более высоких уровней, например TCP. В сетях Frame Relay имеются только постоянные виртуальные каналы, что также упрощает их организацию.

Разработчики технологии Frame Relay сделали важный шаг вперед, предоставив пользователям сети *гарантию пропускной способности* сетевых соединений — свойство, которое до появления Frame Relay не поддерживалось ни одной технологией глобальных сетей с коммутацией пакетов.

Для каждого виртуального соединения в технологии Frame Relay определяется несколько параметров, связанных со скоростью передачи данных.

- **Согласованная скорость передачи данных** (Committed Information Rate, CIR) — гарантированная пропускная способность соединения; фактически сеть гарантирует передачу данных пользователя со скоростью предложенной нагрузки, если эта скорость не превосходит CIR.
- **Согласованная величина пульсации** (Committed Burst Size, Bc) — максимальное количество байтов, которое сеть будет передавать от данного пользователя за интервал времени T, называемый временем пульсации, соблюдая согласованную скорость CIR.
- **Дополнительная величина пульсации** (Excess Burst Size, Be) — максимальное количество байтов, которое сеть будет пытаться передать сверх установленного значения Bc за интервал времени T.

Второй параметр пульсации (Be) позволяет оператору сети дифференцированно обрабатывать кадры, которые не укладываются в профиль CIR. Обычно кадры, которые приводят к превышению пульсации Bc, но не превышают пульсацию Bc + Be, сетью не отбрасываются, а обслуживаются, но без гарантий по скорости CIR. Для запоминания факта нарушения в кадрах Frame Realy имеется специальное поле DE (Discard Eligibility — возможность отбрасывания). В том случае, когда это поле кадра содержит значение 1, последующие коммутаторы данного виртуального канала отбрасывают такой кадр, если испытывают перегрузку. И только если превышен порог Bc + Be, кадры отбрасываются сразу.

Если приведенные величины определены, то время T определяется следующей формулой:

$$T = Bc/CIR.$$

Можно рассматривать значения CIR и T в качестве варьируемых параметров, тогда производной величиной станет пульсация Bc. Обычно для контроля пульсаций трафика выбирается время T, равное 1–2 секунды при передаче компьютерных данных и в диапазоне десятков-сотен миллисекунд при передаче голоса.

Соотношение между параметрами CIR, Bc, Be и T иллюстрирует рис. 19.3 (R — скорость в канале доступа; f_1–f_5 — кадры).

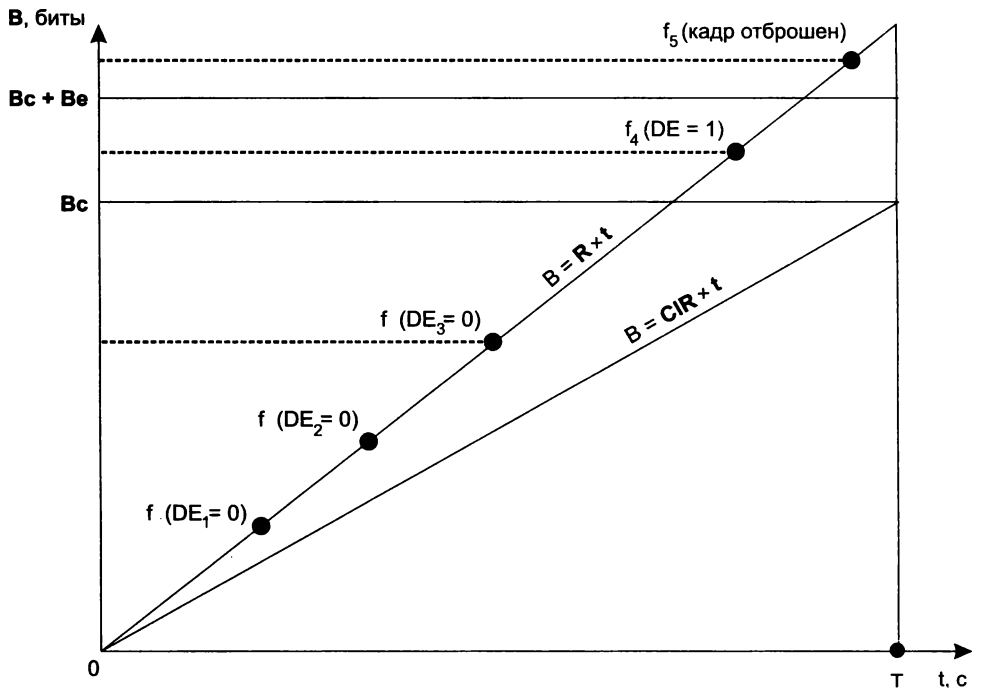


Рис. 19.3. Реакция сети на поведение пользователя

Работа сети описывается двумя линейными функциями, показывающими зависимость количества переданных битов от времени: $B = R \times t$ и $B = CIR \times t$. Средняя скорость поступления данных в сеть составила на этом интервале R бит/с, и она оказалась выше CIR. На рисунке представлен случай, когда за интервал времени T в сеть по виртуальному каналу

поступило 5 кадров. Кадры f_1, f_2 и f_3 доставили в сеть данные, суммарный объем которых не превысил порог Bc, поэтому эти кадры ушли дальше транзитом с признаком DE = 0. Данные кадра f_4, прибавленные к данным кадров f_1, f_2 и f_3, уже превысили порог Bc, но еще не достигли порога Bc + Be, поэтому кадр f_4 также ушел дальше, но уже с признаком DE = 1 (возможно, его удалят последующие коммутаторы). Данные кадра f_5, прибавленные к данным предыдущих кадров, превысили порог Bc + Be, поэтому этот кадр был удален из сети.

На рис. 19.4 приведен пример сети Frame Relay с пятью удаленными региональными отделениями корпорации. Обычно доступ к сети осуществляется по каналам с пропускной способностью, большей чем CIR. Однако при этом пользователь платит не за пропускную способность канала, а за заказанные величины CIR, Bc и Be. Так, при применении в качестве линии доступа канала T-1 и заказа обслуживания со скоростью CIR, равной 128 Кбит/с, пользователь будет платить только за скорость 128 Кбит/с, а скорость канала T-1 в 1,5 Мбит/с окажет влияние на верхнюю границу возможной пульсации Bc + Be.

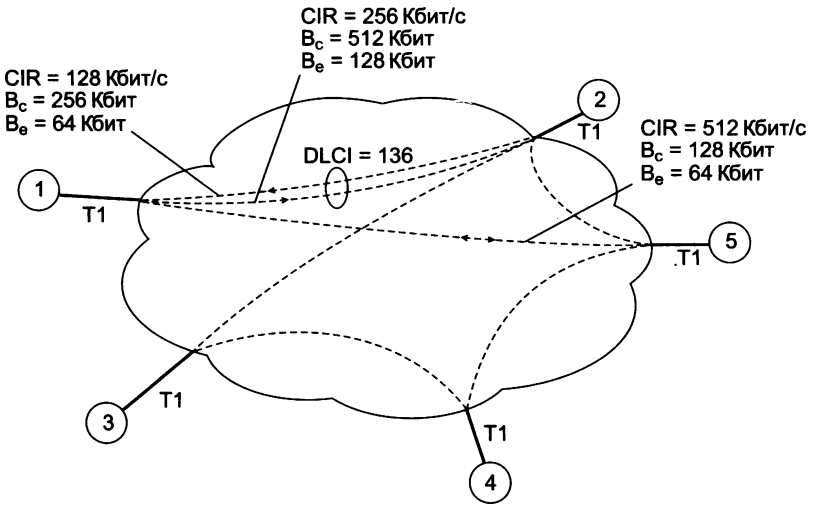


Рис. 19.4. Пример обслуживания в сети Frame Relay

Параметры качества обслуживания могут быть разными для разных направлений виртуального канала. Так, на рисунке абонент 1 соединен с абонентом 2 виртуальным каналом с меткой 136. При направлении от абонента 1 к абоненту 2 канал имеет среднюю скорость 128 Кбит/с с пульсациями Bc = 256 Кбит (интервал T составил 1 с) и Be = 64 Кбит. А при передаче кадров в обратном направлении средняя скорость уже может достигать значения 256 Кбит/с с пульсациями Bc = 512 Кбит и Be = 128 Кбит.

Сети Frame Relay получили большое распространение в 1980-е и в первой половине 1990-х годов. Их услуги с предоставлением гарантий пропускной способности являлись в то время наиболее качественными услугами VPN, и многие корпоративные сети их использовали.

Однако постепенно скорость доступа 2 Мбит/с, которую предоставляли эти сети, становилась явно недостаточной для корпоративных пользователей. К тому же мультимедийный трафик начал все больше интересовать как пользователей, так и провайдеров Интернета,

а сети Frame Relay были рассчитаны только на передачу *компьютерного трафика*. В результате в начале 1990-х годов была начата разработка новой технологии глобальных сетей, получившей название асинхронного режима передачи.

(S) *Технология Frame Relay*

Технология ATM

Асинхронный режим передачи (Asynchronous Transfer Mode, ATM) — это технология, основанная на технике *виртуальных каналов* и предназначенная для использования в качестве единого универсального транспорта сетей с интегрированным обслуживанием. Название этой технологии отражает тот факт, что в ней применяется метод коммутации пакетов, который, как известно, основан на *асинхронном временном мультиплексировании* данных в отличие от синхронного временного мультиплексирования, на котором построены многие технологии коммутации каналов.

Под **интегрированным обслуживанием** здесь понимается способность сети передавать трафик разного типа: *чувствительный к задержкам* (например, голосовой) и *эластичный*, то есть допускающий задержки в широких пределах (например, трафик электронной почты или просмотра веб-страниц). Этим технология ATM *принципиально* отличается от технологии Frame Relay, которая изначально предназначалась только для передачи эластичного компьютерного трафика.

Кроме того, в цели разработчиков технологии ATM входило обеспечение многоуровневой иерархии скоростей и возможности использования первичных сетей SDH для соединения коммутаторов ATM.

В технологии ATM для переноса данных применяются **ячейки**. Принципиально ячейка отличается от кадра только тем, что имеет, во-первых, *фиксированный*, во-вторых, *небольшой* размер.

Длина ячейки составляет 53 байта, а поля данных — 48 байт. Именно такие размеры позволяют сети ATM передавать чувствительный к задержкам аудио- и видеотрафик с необходимым уровнем качества. Небольшой размер ячейки снижает две составляющие задержки: задержку пакетизации и время нахождения ячейки в очереди.

Задержка пакетизации связана с процессом оцифровывания аналоговой (например, голосовой) информации и помещения ее в пакет компьютерной сети. Эту операцию должны выполнять интерфейсные модули коммутаторов ATM, к которым подключены в качестве абонентских устройств обычные аналоговые телефоны. Задержка пакетизации зависит только от размера пакета, так как кодек — устройство, которое выполняет оцифровывание голоса, — работает с постоянной частотой 8 КГц, требуемой для качественного представления голоса в цифровой форме (см. раздел «Дискретизация аналоговых сигналов» в главе 8). Механизм образования этой задержки иллюстрирует рис. 19.5.

На рисунке показан голосовой кодек — устройство, которое представляет голос в цифровой форме. Пусть он выполняет замеры голоса в соответствии со стандартной частотой 8 КГц (то есть через каждые 125 мкс), кодируя каждый замер одним байтом данных. Если мы используем для передачи голоса кадры Ethernet максимального размера, то в один кадр поместится 1500 замеров голоса. В результате первый замер, помещенный в кадр Ethernet,

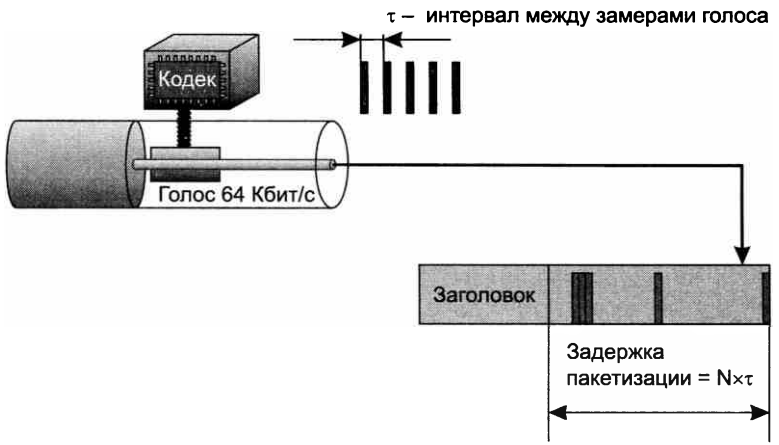


Рис. 19.5. Задержка пакетизации

вынужден будет ждать отправки кадра в сеть (1500 – 1) × 125 = 187 375 мкс, или около 187 мс. Это весьма большая задержка для голосового трафика. Рекомендации стандартов говорят о величине 150 мс как о максимально допустимой *суммарной* задержке голоса, в которую задержка пакетизации входит как одно из слагаемых.

ВНИМАНИЕ

Задержка пакетизации не зависит от битовой скорости протокола, она зависит только от быстродействия кодека и размера поля данных кадра.

Мы знаем, что *время ожидания кадра в очереди* можно сократить, если обслуживать кадры чувствительного к задержкам трафика в приоритетной очереди. Однако если размер кадра может меняться в широком диапазоне, то даже при назначении таким кадрам высшего приоритета время ожидания пакета с замерами голоса в коммутаторе может все равно оказаться недопустимо высоким. Например, пусть пакет с данными в 4500 байт начал передаваться в выходной порт, когда очередь приоритетных голосовых пакетов была пуста. Если скорость интерфейса равна 2 Мбит/с, то время передачи этого пакета займет 18 мс — $(4500 \times 8)/2 \times 10^6 = 0{,}018$. В худшем случае сразу же после начала передачи пакета данных в коммутатор может поступить пакет с замерами голоса. Прерывать передачу пакета в сетях нецелесообразно, так как при распределенном характере сети накладные расходы на оповещение соседнего коммутатора о прерывании пакета, а потом — о возобновлении передачи пакета с прерванного места оказываются слишком большими. Поэтому голосовой пакет будет ждать в очереди 18 мс, пока не завершится передача на линию связи пакета данных, что приведет к значительному снижению качества воспроизведения голоса на приемном конце.

Для поддержания требуемого качества обслуживания и рационального расходования ресурсов в технологии ATM определено пять категорий услуг, которые предназначены для обслуживания различных классов трафика. Классы трафика различаются в зависимости от следующих критериев:

- является ли скорость трафика постоянной (как у голосового трафика) или переменной (как у трафика данных);

- является ли трафик чувствительным к задержкам;
- нужны ли гарантии средней скорости передачи.

Из возможных сочетаний этих свойств трафика (не все сочетания имеют смысл: например, трафик с постоянной скоростью не может не требовать гарантий средней скорости) были отобраны пять и для них созданы отдельные категории услуг.

Очевидно, что сети ATM отличаются от сетей Frame Relay большей степенью соответствия услуг требованиям трафика определенного типа, так как в сетях ATM нужный уровень обслуживания задается не только численными значениями параметров, гарантирующих среднюю скорость передачи данных, но и самой категорией услуги.

Наличие отдельных категорий услуг для наиболее важных классов трафика, таких как чувствительный к задержкам голосовой трафик с постоянной битовой скоростью и чувствительный к задержкам компрессированный видеотрафик с переменной битовой скоростью, сделало ATM гораздо более эффективной технологией мультисервисных сетей, чем технология Frame Relay, которая могла эффективно передавать только нечувствительный к задержкам трафик данных с переменной битовой скоростью.

Технология ATM пережила пик своей популярности во второй половине 1990-х годов, но к настоящему времени она совсем ушла со сцены. Причин отказа от такой, казалось бы, хорошо подходящей для оказания мультисервисных услуг технологии несколько. Одна из них — появление сетей DWDM и рост скорости сетей Ethernet до 1 Гбит/с, а затем и до 10 Гбит/с. Относительно дешевая пропускная способность простой сети Ethernet победила — операторам сетей оказалось гораздо проще предоставлять качественные мультимедийные услуги с помощью недогруженной «простой» сети IP/Ethernet, чем управлять сложной в настройке и эксплуатации сетью IP/ATM. Кроме того, оборудование ATM не смогло перейти порог скорости 622 Мбит/с. Ограничением стал маленький размер ячеек — на высоких скоростях коммутаторы с трудом справляются с обработкой интенсивных потоков таких ячеек.

(S) *Технология ATM*

Технологии двухточечных каналов

В тех случаях, когда IP-маршрутизаторы непосредственно соединены линиями связи физического уровня (кабелями или каналами таких технологий первичных сетей, как PDH, SDH или OTN), функции протокола канального уровня сокращаются по сравнению со случаем, когда на канальном уровне имеется сеть с коммутацией пакетов, например Ethernet или MPLS. Для подобных случаев разработаны специальные протоколы канального уровня с упрощенной функциональностью, которые принято называть двухточечными, или протоколами «точка-точка», что отражает топологию связей между маршрутизаторами.

Протокол HDLC

Протокол HDLC (High-level Data Link Control — высокоуровневое управление линией связи) представляет целое семейство протоколов, реализующих функции канального уровня.

Важным свойством HDLC является его *функциональное разнообразие*. Он может работать в нескольких весьма отличающихся друг от друга режимах, поддерживает не только двухточечные соединения, но и соединения с одним источником и несколькими приемниками, а кроме того, предусматривает различные функциональные роли взаимодействующих станций. Сложность HDLC объясняется тем, что это очень «старый» протокол, разработанный еще в 70-е годы для ненадежных каналов связи. Поэтому в одном из режимов протокол HDLC подобно протоколу TCP поддерживает процедуру установления логического соединения и процедуры контроля передачи кадров, а также восстанавливает утерянные или поврежденные кадры. Существует и дейтаграммный режим работы HDLC, в котором логическое соединение не устанавливается и кадры не восстанавливаются.

В IP-маршрутизаторах чаще всего используется версия протокола HDLC, разработанная компанией Cisco. Несмотря на то что эта версия является фирменным протоколом, она стала стандартом де-факто для IP-маршрутизаторов большинства производителей. Версия Cisco HDLC работает только в дейтаграммном режиме, что соответствует современной ситуации с незашумленными надежными каналами связи. По сравнению со стандартным протоколом версия Cisco HDLC включает несколько расширений, главным из которых является многопротокольная поддержка. Это означает, что в заголовок кадра Cisco HDLC добавлено поле типа протокола, подобное полю EtherType в кадре Ethernet. Это поле содержит код протокола, данные которого переносит кадр Cisco HDLC. В стандартной версии HDLC такое поле отсутствует.

Протокол PPP

Протокол PPP (Point-to-Point Protocol — протокол двухточечной связи) является стандартным протоколом Интернета. Протокол PPP так же, как и HDLC, представляет собой целое семейство протоколов, в которое, в частности, входят:

- протокол управления линией связи (Link Control Protocol, LCP);
- протокол управления сетью (Network Control Protocol, NCP);
- многоканальный протокол PPP (Multi Link PPP, MLPPP);
- протокол аутентификации по паролю (Password Authentication Protocol, PAP);
- протокол аутентификации по квитированию вызова (Challenge Handshake Authentication Protocol, CHAP).

Особенностью протокола PPP, отличающей его от других протоколов канального уровня, является *сложная переговорная процедура* принятия параметров соединения. Стороны обмениваются различными параметрами, такими как качество линии, размер кадров, тип протокола аутентификации и тип инкапсулируемых протоколов сетевого уровня.

В корпоративной сети конечные системы часто отличаются размерами буферов для временного хранения пакетов, ограничениями на размер пакета, списком поддерживаемых протоколов сетевого уровня. Физическая линия, связывающая конечные устройства, может варьироваться от низкоскоростной аналоговой до высокоскоростной цифровой линии с различными уровнями качества обслуживания. Протокол, в соответствии с которым принимаются параметры соединения, называется *протоколом управления линией связи* (LCP).

Чтобы справиться со всеми возможными ситуациями, в протоколе PPP имеется набор стандартных параметров, действующих по умолчанию и учитывающих все стандартные конфигурации. При установлении соединения два взаимодействующих устройства для нахождения взаимопонимания пытаются сначала использовать эти параметры. Каждый конечный узел описывает свои возможности и требования. Затем на основании этой информации принимаются параметры соединения, устраивающие обе стороны. Переговорная процедура протоколов может и не завершиться соглашением о каком-нибудь параметре. Если, например, один узел предлагает в качестве MTU значение 1000 байт, а другой отвергает это предложение и в свою очередь предлагает значение 1500 байт, которое отвергается первым узлом, то по истечении тайм-аута переговорная процедура может закончиться безрезультатно.

Одним из важных параметров соединения PPP является *режим аутентификации*. Для целей аутентификации PPP предлагает по умолчанию *протокол аутентификации по паролю* (PAP), передающий пароль по линии связи в открытом виде, или *протокол аутентификации по квитированию вызова*[1] (CHAP), не передающий пароль по линии связи и поэтому обеспечивающий более высокий уровень безопасности сети. Пользователям также разрешается добавлять новые алгоритмы аутентификации. Кроме того, пользователи могут влиять на выбор алгоритмов сжатия заголовка и данных.

Многопротокольная поддержка — способность протокола PPP поддерживать несколько протоколов сетевого уровня — обусловила распространение PPP как стандарта де-факто. Внутри одного соединения PPP могут передаваться потоки данных различных сетевых протоколов, включая IP, Novell IPX, и многих других, сегодня уже не употребляющихся, а также данные протоколов канального уровня локальной сети.

Каждый протокол сетевого уровня конфигурируется отдельно с помощью соответствующего *протокола управления сетью* (NCP). Под конфигурированием понимается, во-первых, констатация того факта, что данный протокол будет использоваться в текущем сеансе PPP, а во-вторых, переговорное согласование некоторых параметров протокола. Больше всего параметров устанавливается для протокола IP, включая IP-адреса взаимодействующих узлов, IP-адреса DNS-серверов, признак компрессии заголовка IP-пакета и т. д. Для каждого протокола, предназначенного для конфигурирования протокола верхнего уровня, помимо общего названия NCP употребляется особое название, построенное путем добавления аббревиатуры CP (Control Protocol — протокол управления) к имени конфигурируемого протокола: например, для IP — это протокол IPCP, для IPX — IPXCP и т. п.

Под *расширяемостью* протокола PPP понимается как возможность включения новых протоколов в стек PPP, так и возможность применения собственных протоколов пользователей вместо рекомендуемых в PPP по умолчанию. Это позволяет наилучшим образом настроить PPP для каждой конкретной ситуации.

Одной из привлекательных способностей протокола PPP является способность использования нескольких физических линий связи для образования одного логического канала, то есть агрегирование каналов. Эту возможность реализует *многоканальный протокол PPP* (MLPPP).

---

[1] См. раздел «Строгая аутентификация в протоколе CHAP» в главе 27.

Технологии доступа

Проблема последней мили

Организация удаленного доступа является одной из наиболее острых проблем компьютерных сетей. Она получила название *проблемы последней мили*, где под последней милей подразумевается расстояние от точки присутствия (POP) оператора связи до помещений клиентов. Сложность этой проблемы определяется несколькими факторами. С одной стороны, современным пользователям необходим высокоскоростной доступ, обеспечивающий качественную передачу трафика любого типа, в том числе данных, голоса, видео. Для этого нужны скорости в несколько Мбит/с, а для качественного приема телевизионных программ — в несколько десятков Мбит/с. С другой стороны, подавляющее большинство домов в больших и малых городах и особенно в сельской местности по-прежнему соединены с POP абонентскими окончаниями телефонной сети, которые не были рассчитаны на передачу компьютерного трафика.

Кардинальная перестройка кабельной инфраструктуры доступа требует времени — слишком масштабна эта задача из-за огромного количества зданий и домов, географически рассеянных по огромной территории. Процесс прокладки к жилым домам оптического кабеля начался уже давно, но он затронул пока только большие города и крупные здания с множеством потенциальных пользователей.

Долгое время наиболее распространенной технологией доступа был коммутируемый доступ, когда пользователь устанавливал коммутируемое соединение с корпоративной сетью или Интернетом через телефонную сеть с помощью модема, работающего в голосовой полосе частот. Такой способ обладает очевидным и существенным недостатком — скорость доступа ограничена несколькими десятками Кбит/с из-за фиксированной узкой полосы пропускания примерно в 3,4 КГц, выделяемой каждому абоненту телефонной сети (вспомните технику мультиплексирования FDM, применяемую в телефонных сетях и описанную в главе 8). Такие скорости сегодня устраивают все меньшее количество пользователей.

Сегодня существует ряд технологий, способных предоставлять услуги *скоростного удаленного доступа* на основе существующей инфраструктуры абонентских окончаний — телефонных сетей или сетей кабельного телевидения. Эти технологии, обеспечивающие скорость от нескольких сотен Кбит/с до нескольких десятков Мбит/с, используют следующий прием: после достижения POP компьютерные данные уже не следуют по телефонной сети или сети кабельного телевидения, а ответвляются с помощью специального оборудования в сеть передачи данных. Это позволяет преодолеть ограничения на полосу пропускания, отводимую абоненту в телефонной сети или в сети кабельного телевидения, и повысить скорость доступа. Наиболее популярными технологиями такого типа являются технология **ADSL**, использующая телефонные абонентские окончания, и кабельные модемы, работающие поверх сети кабельного телевидения.

Применяются также различные беспроводные технологии доступа, обеспечивающие как фиксированный, так и мобильный доступ. Набор таких беспроводных технологий очень широк, в него входят и беспроводные сети Ethernet (802.11), различные фирменные технологии, передача данных по сети мобильной телефонии, а также технологии фиксированного доступа, например стандарта 802.16.

Рисунок 19.6 иллюстрирует разнообразный и пестрый мир удаленного доступа. Мы видим здесь клиентов различных типов, отличающихся используемым оборудованием и требо-

ваниями к параметрам доступа. Кроме того, помещения клиентов могут быть соединены с ближайшей точкой доступа оператора связи (то есть с ближайшим центральным офисом, если пользоваться терминологией операторов телефонной сети) различными способами: с помощью аналогового или цифрового окончания телефонной сети, телевизионного кабеля, беспроводной связи. Наконец, сам оператор связи может иметь различную специализацию, то есть быть либо поставщиком телефонных услуг, либо поставщиком услуг Интернета, либо оператором кабельного телевидения. Или же он может быть универсальным оператором, предоставляющим весь спектр услуг и обладающим собственными сетями всех типов.

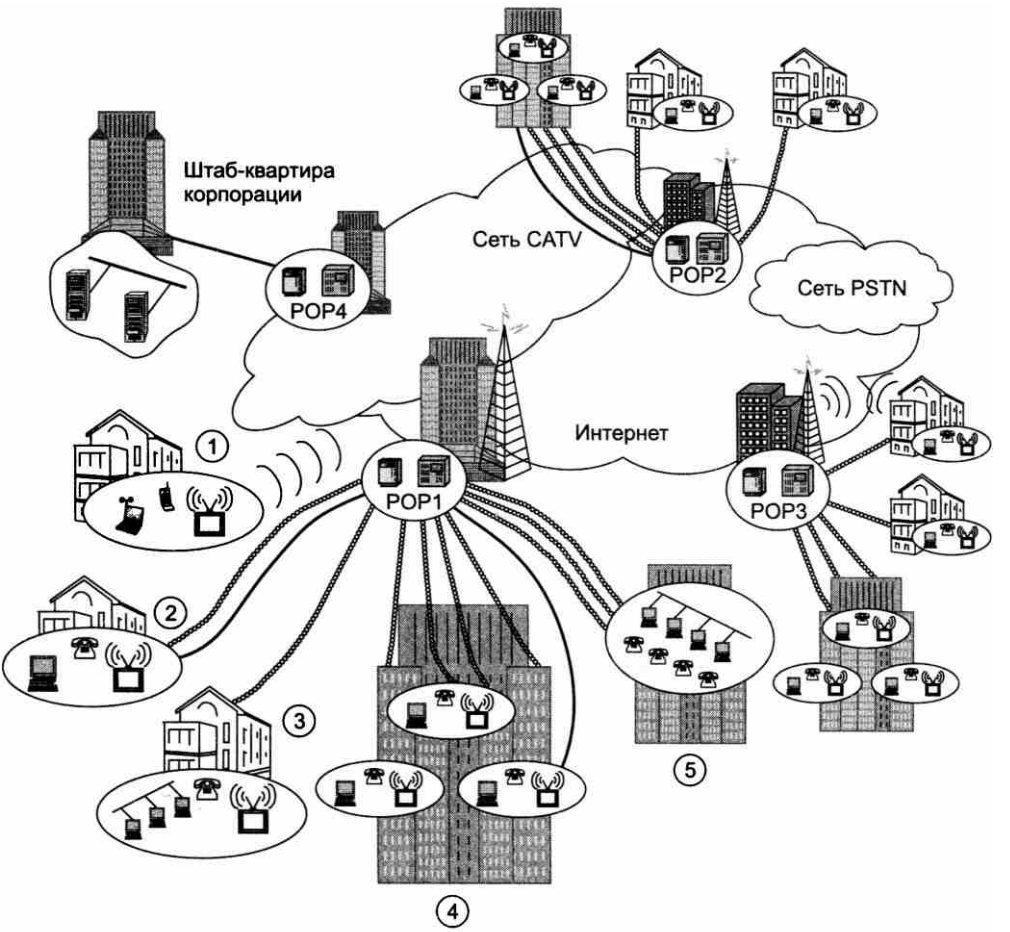


Рис. 19.6. Клиенты удаленного доступа

Коммутируемый аналоговый доступ

Основная идея коммутируемого доступа состоит в том, чтобы использовать имеющуюся телефонную сеть для организации коммутируемого соединения между компьютером домашнего пользователя и **сервером удаленного доступа** (Remote Access Server, **RAS)**,

установленным на границе телефонной и компьютерной сетей. Компьютер пользователя подключается к телефонной сети с помощью **коммутируемого модема**, который поддерживает стандартные процедуры набора номера и имитирует работу телефонного аппарата для установления соединения с RAS.

Схема организации доступа через аналоговую телефонную сеть показана на рис. 19.7.

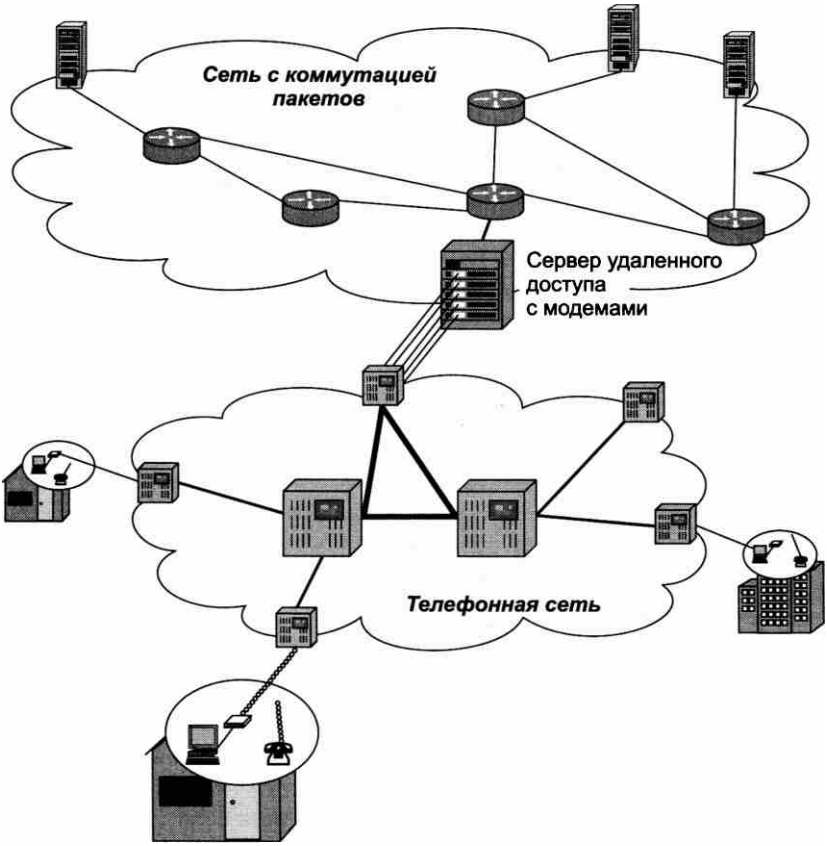


Рис. 19.7. Доступ через телефонную сеть с аналоговыми окончаниями

Сервер RAS имеет два типа соединений: с телефонной сетью через пул модемов и с локальной IP-сетью, соединенной с Интернетом. Для телефонной сети и RAS и модемы клиентов являются обычными пользователями. Коммутаторы телефонной сети сегодня чаще всего цифровые (хотя кое-где остались еще и аналоговые). Однако несмотря на цифровой в основном характер телефонной сети, для использования ее в качестве сети доступа важен тот факт, что ее абонентское окончание является *аналоговым* и между абонентами сети организуется *аналоговый канал с полосой пропускания 4 КГц*.

Для того чтобы получить доступ в Интернет или корпоративную сеть через телефонную сеть, модем пользователя должен выполнить вызов по одному из номеров, присвоенному модемам, находящимся на сервере удаленного доступа. После установления соединения

между модемами в телефонной сети образуется канал с полосой пропускания около 4 КГц. Точное значение ширины имеющейся в распоряжении модемов полосы зависит от типа телефонных коммутаторов на пути от модема пользователя до модема RAS и от поддерживаемых ими сигнальных протоколов. В любом случае, эта полоса не превышает 4 КГц, что принципиально ограничивает скорость передачи данных модемом.

После того как модем установил соединение с RAS, телефонная линия становится недоступной для телефона пользователя, так как модем занимает своим сигналом всю доступную полосу пропускания линии.

Наивысшим достижением современных модемов на канале тональной частоты является скорость в 33,6 Кбит/с, если на пути следования информации приходится выполнять *аналого-цифровое преобразование*, и 56 Кбит/с, если преобразование *цифро-аналоговое*. Такая асимметрия связана с тем, что аналого-цифровое преобразование вносит существенно более значительные искажения в передаваемые дискретные данные, чем цифро-аналоговое.

Очевидно, что такие скорости нельзя назвать приемлемыми для большинства современных приложений, которые широко используют графику и другие мультимедийные формы представления данных.

Модемы RAS обычно устанавливаются в точке присутствия поставщика услуг.

Если же целью пользователя является доступ не в Интернет, а в корпоративную сеть, то он задействует Интернет как промежуточную сеть, которая ведет к корпоративной сети (также подключенной к Интернету). Поскольку плата за доступ в Интернет не зависит от расстояния до узла назначения, удаленный доступ к ресурсам корпорации стал сегодня намного дешевле даже с учетом оплаты за локальный телефонной звонок и доступ в Интернет. Правда, при такой двухступенчатой схеме доступа пользователю приходится выполнять аутентификацию дважды — при доступе к RAS поставщика услуг и при доступе к RAS предприятия. Существуют протоколы, которые исключают подобное дублирование, например **двухточечный протокол туннелирования** (Point-to-Point Tunneling Protocol, PPTP). При работе PPTP сервер удаленного доступа поставщика услуг передает транзитом запрос пользователя серверу аутентификации предприятия и в случае положительного ответа соединяет пользователя через Интернет с корпоративной сетью.

RAS может подключаться к телефонному коммутатору с помощью как аналоговых, так и цифровых окончаний.

Сервер RAS обслуживает подключенные к нему клиентские компьютеры, используя протокол Proxy-ARP (см. главу 14). Это означает, что клиентский компьютер работает в режиме *удаленного узла* локальной IP-сети, с которой соединен сервер RAS, получая на время соединения один из IP-адресов этой сети.

Модемы

Модем реализует функции физического и канального уровней. Канальный уровень нужен модему для того, чтобы выявлять и исправлять ошибки, появляющиеся из-за искажений битов при передаче через телефонную сеть. Вероятность битовой ошибки в этом случае довольно высока, поэтому функция исправления ошибок является очень важной для модема. Для протокола, который работает поверх модемного соединения между удаленным компьютером и RAS, канальный протокол модема прозрачен, его

работа проявляется только в том, что интенсивность битовых ошибок (BER) снижается до приемлемого уровня. Так как в качестве канального протокола между компьютером и RAS сегодня в основном используется протокол PPP, который не занимается восстановлением искаженных и потерянных кадров, способность модема исправлять ошибки оказывается весьма полезной.

Протоколы и стандарты модемов определены в рекомендациях ITU-T серии V и делятся на три группы:

- стандарты, определяющие скорость передачи данных и метод кодирования;
- стандарты исправления ошибок;
- стандарты сжатия данных.

Стандарты метода кодирования и скорости передачи данных. Модемы являются одними из наиболее старых и заслуженных устройств передачи данных; в процессе своего развития они прошли долгий путь, прежде чем научились работать на скоростях до 56 Кбит/с.

Первые модемы работали со скоростью 300 бит/с и исправлять ошибки не умели. Эти модемы функционировали в асинхронном режиме, означающем, что каждый байт передаваемой компьютером информации передавался асинхронно по отношению к другим байтам, для чего он сопровождался стартовыми и стоповыми символами, отличающимися от символов данных. Асинхронный режим упрощает устройство модема и повышает надежность передачи данных, но существенно снижает скорость передачи, так как каждый байт дополняется одним или двумя избыточными старт-стопными символами.

Современные модемы могут работать как в асинхронном, так и в синхронном режиме.

Переломным моментом в истории развития модемов стало принятие **стандарта V.34**, который повысил максимальную скорость передачи данных в два раза, с 14 до 28 Кбит/с по сравнению со своим предшественником — стандартом V.32. Особенностью стандарта V.34 являются *процедуры динамической адаптации* к изменениям характеристик канала во время обмена информацией. В V.34 определено 10 согласительных процедур, по которым модемы после тестирования линии выбирают свои основные параметры: несущую полосу и полосу пропускания, фильтры передатчика и др. Адаптация осуществляется в ходе сеанса связи без прекращения и без разрыва установленного соединения. Возможность такого адаптивного поведения была обусловлена развитием техники интегральных схем и микропроцессоров. Первоначальное соединение модемов проводится по стандарту V.21 на минимальной скорости 300 бит/с, что позволяет работать на самых плохих линиях. Затем модемы продолжают переговорный процесс до тех пор, пока не достигают максимально возможной в данных условиях производительности. Применение адаптивных процедур сразу позволило поднять скорость передачи данных более чем в два раза по сравнению с предыдущим стандартом — V.32 bis.

Принципы адаптивной настройки к параметрам линии были развиты в **стандарте V.34+**. Стандарт V.34+ позволил несколько повысить скорость передачи данных за счет усовершенствования метода кодирования. Один передаваемый кодовый символ несет в новом стандарте в среднем не 8,4 бита, как в протоколе V.34, а 9,8. При максимальной скорости передачи кодовых символов в 3429 бод (это ограничение преодолеть нельзя, так как оно определяется полосой пропускания канала тональной частоты) усовершенствованный метод кодирования дает скорость передачи данных в 33,6 Кбит/с ($3429 \times 9{,}8 = 33\,604$).

Протоколы V.34 и V.34+ позволяют работать на двухпроводной выделенной линии в дуплексном режиме. Дуплексный режим передачи в стандартах V.34, V.34+ поддерживается не частотным разделением канала, а одновременной передачей данных в обоих направлениях. Принимаемый сигнал определяется вычитанием с помощью процессоров DSP передаваемого сигнала из общего сигнала в канале. Для этой операции используются также процедуры эхоподавления, так как передаваемый сигнал, отражаясь от ближнего и дальнего концов канала, вносит искажения в общий сигнал.

ПРИМЕЧАНИЕ

Заметьте, что метод передачи данных, описанный в проекте стандарта 802.3ab, определяющего работу технологии Gigabit Ethernet на витой паре категории 5, взял многое из стандартов V.32–V.34+.

Стандарт V.90 описывает технологию недорогого и быстрого доступа пользователей к сетям поставщиков услуг. Этот стандарт предлагает асимметричный обмен данными: со скоростью до 56 Кбит/с из сети и со скоростью до 33,6 Кбит/с в сеть. Стандарт совместим со стандартом V.34+. Именно этот стандарт имелся в виду, когда мы говорили о возможности нисходящей передачи данных со скоростью 56 Кбит/с при условии, что вдоль всего пути не встретится ни одного аналого-цифрового преобразователя.

В **стандарте V.92** учитывается возможность принятия модемом второго вызова во время соединения. В таких случаях современные станции передают на телефонный аппарат специальные двойные тоновые сигналы, так что абонент может распознать эту ситуацию и, нажав на аппарате кнопку Flash, переключиться на второе соединение, переведя первое соединение в режим удержания. Модемы предыдущих стандартов в таких случаях просто разрывают соединение, что не всегда удобно для абонента, — может быть, в этот момент он заканчивает загружать из Интернета большой файл, и вся его работа пропадает.

Коррекция ошибок. Для модемов, работающих с DTE по асинхронному интерфейсу, комитет ITU-T разработал **протокол коррекции ошибок V.42**. До его принятия в модемах, работающих по асинхронному интерфейсу, коррекция ошибок обычно выполнялась по фирменным протоколам Microcom. Эта компания реализовала в своих модемах несколько разных процедур коррекции ошибок, назвав их сетевыми протоколами Microcom (Microcom Networking Protocol, MNP) классов 2–4.

В стандарте V.42 основным является **протокол доступа к линии связи для модемов** (Link Access Protocol for Modems, LAP-M). Рекомендации V.42 позволяют устанавливать связь без ошибок с любым модемом, поддерживающим этот стандарт, а также с любым MNP-совместимым модемом. Протокол LAP-M принадлежит семейству HDLC и в основном работает так же, как и другие протоколы этого семейства, — с установлением соединения, кадрированием данных, нумерацией кадров и восстановлением кадров с поддержкой метода скользящего окна.

Сжатие данных. Почти все современные модемы при работе по асинхронному интерфейсу поддерживают **стандарты сжатия данных ITU-T V.42bis** и **MNP-5** (обычно с коэффициентом 1:4, некоторые модели — до 1:8). Сжатие данных повышает пропускную способность линии связи. Передающий модем автоматически сжимает данные, а принимающий их — восстанавливает. Модем, поддерживающий протокол сжатия, всегда пытается установить связь со сжатием данных, но если второй модем этот протокол не поддерживает, то и первый модем переходит на обычную связь без сжатия.

При работе модемов по синхронному интерфейсу наиболее популярным является протокол **сжатия синхронных потоков данных** (Synchronous Data Compression, SDC) компании Motorola.

(S) *Коммутируемый доступ через аналоговую телефонную сеть*

Коммутируемый доступ через сеть ISDN

Целью создания технологии **ISDN** (Integrated Services Digital Network — **цифровая сеть с интегрированным обслуживанием**) было построение всемирной сети, которая должна была прийти на смену телефонной сети и, будучи такой же доступной и распространенной, предоставлять миллионам своих пользователей разнообразные услуги, как телефонные, так и передачи данных. Передача телевизионных программ по ISDN не предполагалась, было решено ограничиться пропускной способностью абонентского окончания для массовых пользователей в 128 Кбит/с и 2 Мбит/с для корпоративных пользователей.

По многим причинам внедрение ISDN происходило очень медленно. Главным препятствием на пути распространения ISDN стала необходимость организации **цифрового абонентского окончания** (Digital Subscriber Line, DSL), требующего модернизации миллионов абонентских окончаний. В результате процесс, который начался в 80-е годы, растянулся больше чем на десять лет, так что к моменту появления в домах пользователей в 90-е годы абонентских окончаний ISDN услуги этой сети просто морально устарели. Скорость доступа 128 Кбит/с уже тогда была недостаточной для многих пользователей, а услуги со скоростью 2 Мбит/с были очень дорогими. В результате ISDN осталась нишевой технологией, используемой все реже и реже.

Базовой скоростью сети ISDN является скорость канала DS-0, то есть 64 Кбит/с. Эта скорость ориентируется на самый простой метод кодирования голоса — PCM, хотя дифференциальное кодирование и позволяет передавать голос с тем же качеством на скорости 32 или 16 Кбит/с.

> Одной из оригинальных идей, положенных в основу ISDN, является совместное использование принципов коммутации каналов и пакетов.

Однако сеть с коммутацией пакетов, работающая в составе ISDN, выполняет только служебные функции — с ее помощью передаются сообщения сигнального протокола. А вот основная информация, то есть сам голос, по-прежнему передается через сеть с коммутацией каналов. В таком разделении функций есть вполне понятная логика — сообщения о вызове абонентов образуют пульсирующий трафик, поэтому его эффективнее передавать по сети с коммутацией пакетов.

Пользовательский интерфейс абонента ISDN основан на каналах трех типов: B, D и H.

Каналы типа B обеспечивают передачу пользовательских данных (оцифрованного голоса, компьютерных данных или смеси голоса и данных). Каналы типа B могут иметь постоянные соединения, а также образовывать так называемые полупостоянные соединения, которые эквивалентны соединениям каналов обычной телефонной сети.

Канал типа D является каналом доступа к служебной сети с коммутацией пакетов, передающей сигнальную информацию со скоростью 16 или 64 Кбит/с. Передача адресной ин-

формации, на основе которой осуществляется коммутация каналов типа B в коммутаторах сети, является основной функцией канала D. Другой его функцией является поддержание сервиса низкоскоростной сети с коммутацией пакетов для пользовательских данных. Обычно этот сервис выполняется сетью в то время, когда каналы типа D свободны от выполнения основной функции.

Пользовательский интерфейс ISDN представляет собой набор логических каналов определенного типа и с определенными скоростями. Сеть ISDN поддерживает два вида пользовательского интерфейса: с начальной (Basic Rate Interface, BRI) и основной (Primary Rate Interface, PRI) скоростями передачи данных.

Начальный интерфейс (BRI) ISDN предоставляет пользователю два канала по 64 Кбит/с для передачи данных (каналы типа B) и один канал с пропускной способностью 16 Кбит/с для передачи управляющей информации (канал типа D). По замыслу разработчиков технологии ISDN, один канал типа B пользователь может задействовать для подключения цифрового телефона, а второй — для подключения компьютера.

Основной интерфейс (PRI) ISDN предназначен для пользователей с повышенными требованиями к пропускной способности сети. Он обеспечивает скорость 2,048 Мбит/с (европейский вариант) или 1,544 Мбит/с (американский вариант).

Несмотря на значительные отличия от аналоговых телефонных сетей, сети ISDN используются для организации удаленного доступа в основном так же, как аналоговые телефонные сети, то есть как сети с коммутацией каналов, но только более скоростные. Кроме того, качество цифровых каналов гораздо выше, чем аналоговых, а следовательно, существенно ниже процент искаженных кадров и значительно выше полезная скорость обмена данными.

Обычно интерфейс BRI служит в коммуникационном оборудовании для подключения отдельных компьютеров или небольших локальных сетей домашних пользователей, а интерфейс PRI — для подключения сети средних размеров с помощью маршрутизатора.

Схема удаленного доступа через ISDN показана на рис. 19.8.

Для удаленного доступа необходимо оснастить компьютеры пользователей терминальными адаптерами, а в POP установить маршрутизатор, имеющий один или несколько интерфейсов PRI. В этом случае максимальная скорость доступа для отдельного пользователя будет равна скорости передачи двух каналов типа B, то есть 128 Кбит/с. Драйверы терминальных адаптеров ISDN умеют объединять два отдельных физических канала типа B в один логический канал. Для этого служит расширение протокола PPP — многоканальный протокол MLPPP. Если пользователь удаленного доступа согласен ограничиться скоростью 64 Кбит/с, он может задействовать второй канал типа B своего интерфейса BRI для параллельной работы телефона ISDN, что невозможно сделать при применении аналогового коммутируемого модема.

(S) *Сети ISDN*

Технология ADSL

Технология **асимметричного цифрового абонентского окончания** (Assymetric Digital Subscriber Line, **ADSL**) была разработана для обеспечения скоростного доступа в Интернет массовых индивидуальных пользователей, квартиры которых оснащены обычными абонентскими телефонными окончаниями. Появление технологии ADSL было револю-

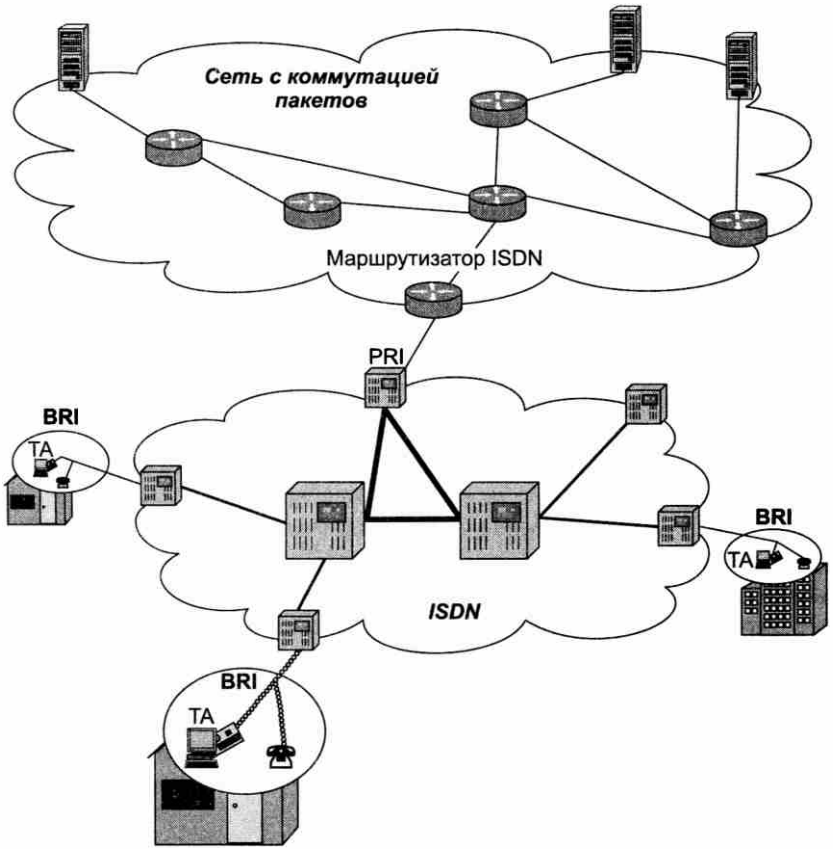


Рис. 19.8. Удаленный доступ с использованием ISDN

ционным событием для массовых пользователей Интернета, потому что для них оно означало повышение скорости доступа в десятки (а то и более) раз без какого бы то ни было изменения кабельной проводки в квартире и в доме.

Для доступа через ADSL, так же как и для аналогового коммутируемого доступа, нужны телефонные абонентские окончания и модемы.

> Принципиальным отличием доступа через ADSL от коммутируемого доступа является то, что ADSL-модемы работают только в пределах абонентского окончания, в то время как коммутируемые модемы используют возможности телефонной сети, устанавливая в ней соединение «из конца в конец», которое проходит через несколько транзитных коммутаторов.

Поэтому если традиционные телефонные модемы (например, V.34, V.90) должны обеспечивать передачу данных на канале с полосой пропускания в 3100 Гц, то ADSL-модемы получают в свое распоряжение полосу порядка 1 МГц — эта величина зависит от длины кабеля, проложенного между помещением пользователя и POP, и сечения проводов этого кабеля.

Схема доступа через ADSL показана на рис. 19.9.

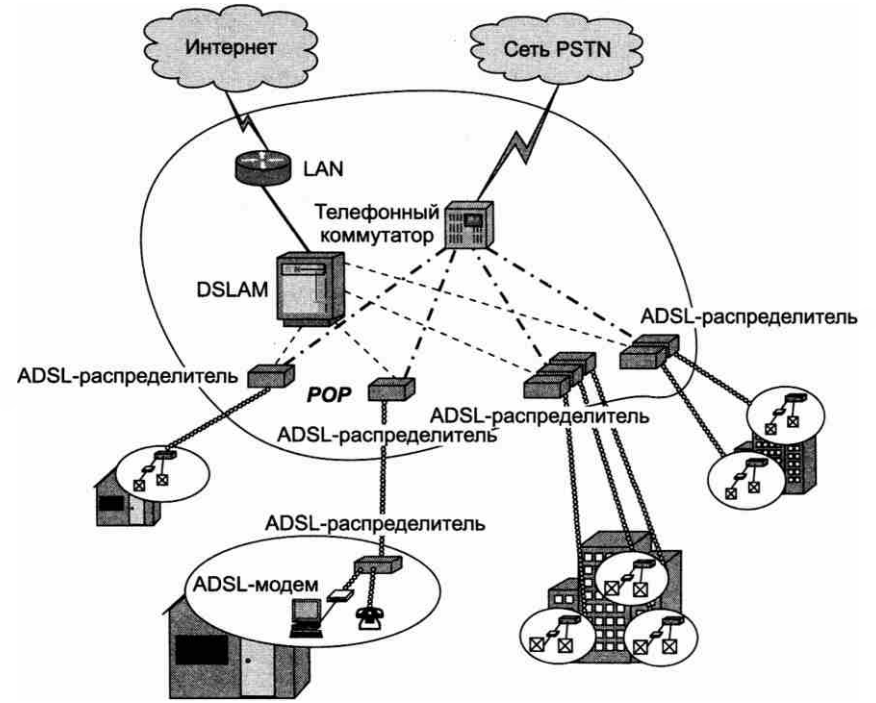


Рис. 19.9. Отличия условий работы ADSL-модемов от обычных модемов

ADSL-модемы, подключаемые к обоим концам короткой линии между абонентом и POP, образуют три канала: высокоскоростной нисходящий канал передачи данных из сети в компьютер, менее скоростной восходящий канал передачи данных из компьютера в сеть и канал телефонной связи, по которому передаются обычные телефонные разговоры. Передача данных в канале от сети к абоненту в стандарте ADSL 1998 года происходит со скоростью от 1,5 до 8 Мбит/с, а в канале от абонента к сети — от 16 Кбит/с до 1 Мбит/с; для телефона оставлена традиционная полоса в 4 КГц (рис. 19.10).

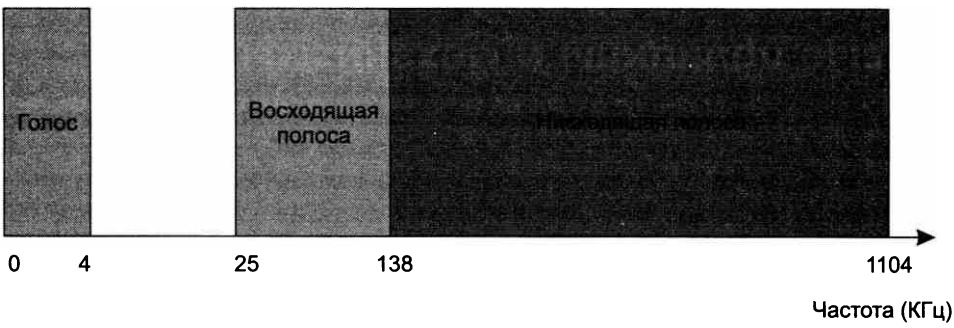


Рис. 19.10. Распределение полосы пропускания абонентского окончания между каналами ADSL

Для асимметрии нисходящей и восходящей скоростей полоса пропускания абонентского окончания делится между каналами также асимметрично. На рис. 19.10 показано распределение полосы между каналами, при этом приведенные величины для восходящей и нисходящей полос являются максимальными значениями, которые модем в каждом конкретном сеансе может использовать полностью или же частично.

Неопределенность используемых полос частот объясняется тем, что модем постоянно тестирует качество сигнала и выбирает только те части выделенного для передачи спектра, в которых соотношение сигнал/шум является приемлемым для устойчивой передачи дискретных данных. Заранее сказать, в каких частях выделенного спектра это соотношение окажется приемлемым, невозможно, так как это зависит от длины абонентского окончания, сечения провода, качества витой пары в целом, а также помех, которые наводятся на провода абонентского окончания. ADSL-модемы умеют адаптироваться к качеству абонентского окончания и выбирать максимально возможную на данный момент скорость передачи данных.

В помещении клиента устанавливается распределитель, который выполняет разделение частот между ADSL-модемом и обычным аналоговым телефоном, обеспечивая их совместное сосуществование.

В POP устанавливается так называемый **мультиплексор доступа к цифровому абонентскому окончанию** (Digital Subscriber Line Access Multiplexer, **DSLAM**). Он принимает компьютерные данные, отделенные распределителями на дальнем конце абонентских окончаний от голосовых сигналов. DSLAM-мультиплексор должен иметь столько ADSL-модемов, сколько пользователей удаленного доступа обслуживает поставщик услуг с помощью телефонных абонентских окончаний.

После преобразования модулированных сигналов в дискретную форму DSLAM отправляет данные на IP-маршрутизатор, который также обычно находится в помещении POP. Далее данные поступают в магистраль передачи данных поставщика услуг и доставляются в соответствии с IP-адресами назначения на публичный сайт Интернета или в корпоративную сеть пользователя. Отделенные распределителем голосовые сигналы передаются на телефонный коммутатор, который обрабатывает их так, как если бы абонентское окончание пользователя было непосредственно к нему подключено.

Широкое распространение технологий ADSL должно сопровождаться некоторой перестройкой работы поставщиков услуг Интернета и операторов телефонных сетей, так как их оборудование должно теперь работать совместно. Возможен также вариант, когда альтернативный оператор связи берет оптом в аренду большое количество абонентских окончаний у традиционного местного оператора или же арендует некоторое количество модемов в DSLAM.

Технология ADSL постоянно совершенствуется, стараясь поднять потолки скоростей нисходящего и восходящего потоков в соответствии с растущими требованиями пользователей, желающих смотреть видео в хорошем качестве на домашних компьютерах. В стандарте ADSL2+ удалось поднять верхний потолок скорости нисходящего потока до 24, а восходящего — до 1,4 Мбит/с за счет расширения используемой полосы пропускания абонентского окончания до 2,2 МГц и более эффективного использования этой полосы.

Высокие скорости ADSL-модемов порождают для поставщиков услуг новую проблему, а именно проблему дефицита пропускной способности. Действительно, если бы каждый абонент доступа через ADSL загружал данные из Интернета с максимальной скоростью, например 1 Мбит/с, то при 100 абонентах поставщику услуг потребовался бы канал

с пропускной способностью 100 Мбит/с, то есть Fast Ethernet, а если разрешить пользователям работать со скоростью 6 Мбит/с, то уже нужен канал ATM 622 Мбит/с или Gigabit Ethernet. Для обеспечения необходимой скорости многие устройства DSLAM имеют встроенный коммутатор ATM или Gigabit Ethernet. Технология ATM привлекает разработчиков DSLAM не только своей высокой скоростью, но и тем, что она ориентирована на соединение. При применении сети ATM на канальном уровне компьютер пользователя перед передачей данных должен обязательно установить соединение с сетью поставщика услуг. Это дает возможность контролировать доступ пользователей и учитывать время работы и объем переданных данных, если при оплате за услугу эти параметры учитываются.

Технология SDSL позволяет на одной паре абонентского окончания организовать два симметричных канала передачи данных. Канал тональной частоты в этом случае не предусматривается. Обычно скорости каналов в восходящем и нисходящем направлениях составляют по 2 Мбит/с, но как и у технологии ADSL, эта скорость зависит от качества линии и расстояния до оборудования DSLAM. Технология SDSL разработана в расчете на небольшие офисы, локальные сети которых содержат собственные источники информации, например веб-сайты или серверы баз данных. Поэтому характер трафика здесь ожидается скорее симметричный, так как доступ через SDSL требуется не только к внешним сетям из локальных сетей, но и к таким источникам информации извне. В технологии SDSL используется также голосовая часть спектрального диапазона, поэтому при работе SDSL-модема нельзя параллельно с передачей данных разговаривать по обычному телефону, как это допускается при работе ADSL-модема.

Широкое применение доступа через xDSL наносит еще один удар технологии ISDN. При применении этого типа абонентских окончаний пользователь получает еще и интегрированное обслуживание двух сетей: телефонной и компьютерной. Но для пользователя наличие двух сетей оказывается незаметным, для него ясно только то, что он может одновременно пользоваться обычным телефоном и подключенным к Интернету компьютером. Скорость же компьютерного доступа при этом превосходит возможности интерфейса PRI сети ISDN при существенно более низкой стоимости, определяемой низкой стоимостью инфраструктуры IP-сетей.

Пассивные оптические сети

Проведение оптического волокна от точки присутствия оператора до здания пользователя является самым качественным решением проблемы организации удаленного доступа, так как позволяет обеспечить высокие скорости обмена данными и хорошую защищенность данных. Для подключения многочисленных зданий индивидуальных и корпоративных пользователей, занимающих значительную территорию, оператор должен создать некоторую сеть доступа. Для ее организации оператор может задействовать технологии PDH, SDH или OTN, которые мы рассматривали в главе 10. Оптическая сеть, построенная на этих технологиях, должна включать такие устройства, как усилители, повторители и мультиплексоры. Все эти устройства являются активными, то есть включают электрические схемы, нуждающиеся в электропитании. Такое решение, безусловно, вполне допустимо, и многие операторы его применяют для построения сетей доступа, требующих прокладки оптического волокна до помещений пользователей. Однако это решение является довольно дорогим.

Для удешевления оптической сети доступа еще в середине 90-х годов была предложена технология, получившая название **пассивная оптическая сеть** (Passive Optical Network, **PON**).

Название «пассивная» происходит от применения в сети пассивных оптических устройств — разветвителей (splitter), не требующих электропитания. С помощью разветвителей организуется древовидная оптоволоконная структура, соединяющая точку присутствия оператора с помещениями пользователей (рис. 19.11). Разветвитель выполняет простую операцию — он направляет световой сигнал из одного входного волокна в несколько выходных, идущих по направлению к пользователям. Разветвитель выполняет также обратную операцию мультиплексирования сигналов пользователей в одно волокно, идущее к POP.

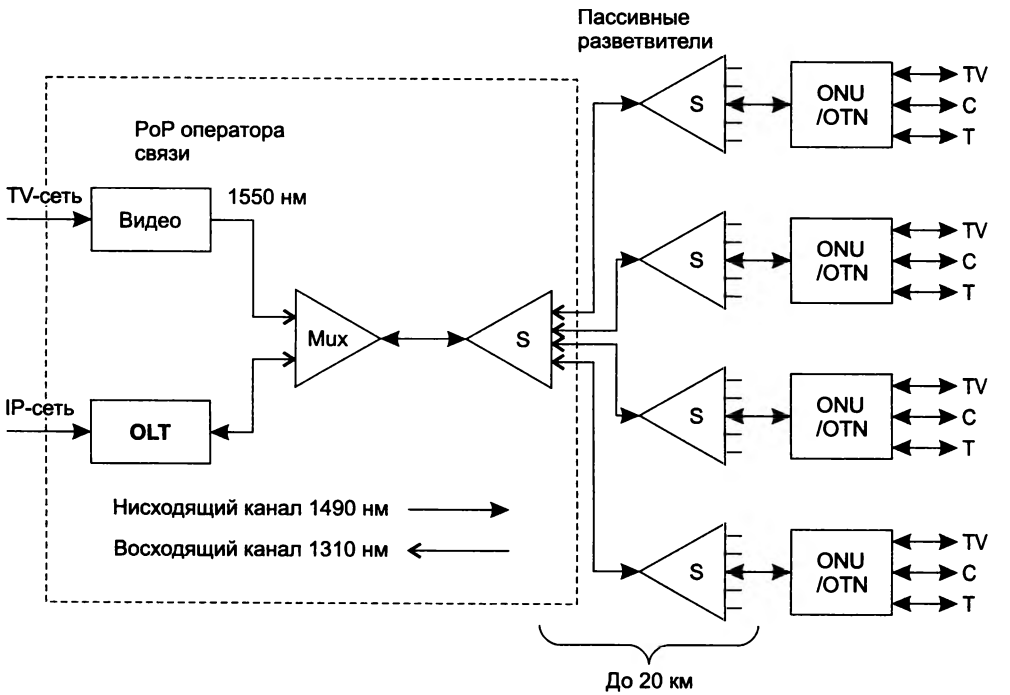


Рис. 19.11. Структура пассивной оптической сети доступа

Коэффициент разветвления этих пассивных устройств может достигать значений 1:16 или 1:32, а максимальное расстояние от них до активных устройств, находящихся в точке присутствия оператора, равно 20 км (применяется одномодовое волокно, то есть SMF). Поскольку разветвители не требуют электропитания, их можно размещать в близких к зданиям пользователей точках городской инфраструктуры, где активные устройства не смогли бы работать, тем самым сокращается суммарная длина оптического волокна. Экономия достигается за счет того, что вместо N индивидуальных волокон на отрезке между центральным офисом оператора и разветвителем требуется только одно общее волокно. Кроме того, пассивные разветвители сами по себе дешевле активных мультиплексоров. Оператор может применять не один, а два уровня разветвления. На рисунке показан имен-

но такой вариант: первый разветвитель распределяет сигнал между четырьмя волокнами, а разветвители второго уровня распределяют его между конечными пользователями.

К вершине дерева подключается центральный **терминал оптической линии** (Optical Line Terminal, **OLT**). К каждому оптическому волокну на стороне пользователя подключается **модуль оптической сети** (Optical Network Unit, **ONU**), совместно с OLT организующий прием и передачу данных между пользователем и сетью оператора связи. Кроме ONU на стороне пользователя должен работать **терминал оптической сети** (Optical Network Terminal, **ONT**), который обеспечивает интерфейсы для терминальных устройств пользователя — телевизора, компьютера и телефона (соответственно TV, C и T на рисунке). Часто функции ONU и ONT совмещены в одном и том же устройстве (как это и показано на рисунке).

Для передачи цифровой информации в нисходящем (от OLT к модулям ONU) и восходящем (от модулей ONU к OLT) направлениях используется две волны, распространяемые в одном волокне: 1490 нм для нисходящего направления и 1310 нм для восходящего. Для передачи видеосигнала обычно выделяется отдельная волна в 1550 нм, идущая в нисходящем направлении. Волны мультиплексируются в OLT и ONU по технологии WDM.

По своей природе древовидная сеть доступа, построенная на пассивных разветвителях и отрезках оптического волокна, является *разделяемой средой*. Действительно, световой сигнал некоторой волны, отправленный OLT, одновременно распространяется по всем отрезкам оптического волокна и достигает всех пользователей сети. При передаче в обратном направлении разделяемой средой являются отрезки волокна от разветвителей, установленных на стороне пользователей, а также от разветвителя, установленного в POP (в нашем примере имеется четыре разделяемых среды для обратного направления).

Очевидно, что для работы на разделяемой среде необходим какой-то способ доступа, регулирующий ее использование таким образом, чтобы сигналы, посылаемые разными узлами, не смешивались. Правда, проблема скоординированного применения разделяемой среды существует только для восходящего направления. Для нисходящего направления в сети имеется только один передатчик — OLT. Поэтому передатчик OLT передает кадры данных (например, кадры Ethernet), направленные некоторому конечному узлу сети PON, тогда, когда ему это необходимо (то есть тогда, когда такой кадр поступает в OLT из локальной сети оператора, к которой подключен передатчик). Кадр поступает по древовидной пассивной оптической сети *на все* конечные узлы сети, но принимает его только тот узел, который распознает собственный адрес в заголовке кадра, остальные узлы просто игнорируют чужой кадр.

Для восходящего направления обычно применяется схема с центральным арбитром в сочетании с мультиплексированием с разделением времени TDM. Арбитром является центральное устройство OLT, оно управляет распределением тайм-слотов между модулями ONU. Модуль ONU передает в восходящем направлении кадры данных только в пределах своего тайм-слота, все остальное время он простаивает, накапливая кадры для передачи в своем буфере. Алгоритм распределения тайм-слотов между модулями ONU может быть адаптивным, подстраивающимся под имеющиеся потребности модулей ONU в передаче кадров.

Для телевизионного сигнала проблемы разделения среды не существует, так как он передается только в нисходящем направлении, причем всем приемникам нужен один и тот же сигнал.

Как и во всех технологиях, использующих разделяемую среду, пропускная способность сети PON, приходящаяся на один узел, может быть существенно меньше, чем скорость передачи данных в среде. Если все узлы сети активно обмениваются информацией с внешним миром, то доля скорости для узла снижается в N раз.

Существует две группы стандартов PON: от ITU-T и от IEEE. Последние версии этих стандартов поддерживают скорости передачи данных 1 и 10 Гбит/с.

Стандарты ITU-T GPON (Gigabit PON) и XG-PON (10 Gigabit PON) обеспечивают совместимость с технологией SDH. Эти стандарты предлагают собственный формат кадров, который может эффективно переносить несколько пользовательских кадров, например кадров Ethernet. Кадры GPON и XG-PON также могут переносить данные SDH с сохранением их синхронности, что важно при передаче голоса и видео. Стандарты ITU-T обеспечивают несимметричные скорости передачи данных: 2,488/1,244 и 9,953/2,488 Гбит/с.

Стандарты IEEE EPON (Ethernet PON, 802.3ah) и 10G-EPON (10G Ethernet PON, 802.3av) поддерживают кадры Ethernet непосредственно. В этих стандартах канал передачи данных является симметричным, то есть данные передаются как в нисходящем, так и в восходящем направлении с одинаковой скоростью 1 и 10 Гбит/с соответственно.

(S) *Доступ через сети кабельного телевидения*

(S) *Беспроводной доступ*

Выводы

Техника виртуальных каналов дает оператору сети большую степень контроля над путями прохождения данных, чем техника дейтаграммной передачи данных, применяемая в таких технологиях, как IP и Ethernet. По этой причине в большинстве технологий канального уровня, разработанных специально для глобальных сетей, таких как Frame Relay и ATM, используется техника виртуальных каналов.

Сети Frame Relay работают на основе постоянных виртуальных каналов. Эти сети позволяют передавать компьютерный трафик с гарантиями его средней скорости и объема пульсации.

Технология ATM является дальнейшим развитием идей предварительного резервирования пропускной способности виртуального канала, реализованных в технологии Frame Relay. Технология ATM предоставляет пользователям услуги различных категорий, ориентированные на эффективную передачу основных классов трафика — голосового, видео и данных. Несмотря на тонкие механизмы обеспечения качества обслуживания при передаче мультимедийного трафика, технология ATM не выдержала конкуренции с технологией Ethernet, обеспечившей высокое качество обслуживания трафика разного типа за счет повышения пропускной способности сети до гигабитных скоростей.

Организация удаленного доступа является одной из наиболее острых проблем компьютерных сетей, так как большинство помещений массовых пользователей по-прежнему оснащено только окончаниями телефонных сетей, плохо приспособленных для высокоскоростной передачи компьютерных данных.

Технология ADSL решает проблему удаленного доступа за счет полного использования полосы пропускания «последней мили».

Технология пассивных оптических сетей позволяет экономично довести оптические окончания до помещений массовых пользователей.

Контрольные вопросы

1. Уникальность метки виртуального канала должна быть обеспечена в пределах:
   а) сети данного провайдера;
   б) отдельного коммутатора сети;
   в) порта отдельного коммутатора сети.
2. В соглашении SLA между клиентом и поставщиком услуг Frame Relay оговаривается значение CIR = 512 Кбит/с на периоде 100 мс, при этом при подсчете скорости учитывается только поле данных кадров Frame Relay. Пусть на очередном периоде 100 мс пограничный коммутатор клиента послал в сеть 7 кадров с размерами поля данных 1000, 1500, 1200, 1500, 1000, 1300 и 1500 байт соответственно. Были ли эти кадры помечены пограничным коммутатором провайдера признаком DE = 1, и если да, то какие?
3. Задержка пакетизации — это:
   а) время передачи пакета в линию связи;
   б) время между помещением в пакет первого и последнего замеров голоса;
   в) время ожидания пакета в очереди к выходному интерфейсу.
4. За счет чего скорость доступа в технологии ADSL намного выше, чем при доступе через телефонную аналоговую сеть с помощью модемов стандарта V.90?
5. Является ли инфраструктура оптических линий связи технологии PON разделяемой средой?

ГЛАВА 20 Технология MPLS

Базовые принципы и механизмы MPLS

Технология **многопротокольной коммутации по меткам** (MultiProtocol Label Switching, **MPLS**) считается многими специалистами одной из самых перспективных транспортных технологий. Эта технология объединяет технику виртуальных каналов с функциональностью стека TCP/IP.

Главное достоинство MPLS видится сегодня многими специалистами в способности предоставлять разнообразные транспортные услуги в IP-сетях, в первую очередь — услуги виртуальных частных сетей. Эти услуги отличаются разнообразием, они могут предоставляться как на сетевом, так и на канальном уровне. Кроме того, MPLS дополняет дейтаграммные IP-сети таким важным свойством, как передача трафика в соответствии с техникой виртуальных каналов, что позволяет выбирать нужный режим передачи трафика в зависимости от требований услуги. Виртуальные каналы MPLS обеспечивают инжиниринг трафика, так как они поддерживают детерминированные маршруты.

Совмещение коммутации и маршрутизации

Технология MPLS объединяет в одном коммуникационном устройстве два метода продвижения пакетов — дейтаграммный метод и метод коммутации виртуальных каналов.

Дейтаграммное продвижение реализуется протоколом IP — он работает точно так же, как и в традиционном IP-маршрутизаторе, при этом таблица маршрутизации может создаваться как вручную, так и протоколами маршрутизации стека TCP/IP.

В то же время в этом коммуникационном устройстве, называемом **маршрутизатором с коммутацией по меткам** (Label Switch Router, **LSR**)[1], имеется второй модуль продвижения, работающий в соответствии с техникой коммутации виртуальных каналов, который здесь называется модулем **коммутации по меткам** (рис. 20.1).

Оба модуля продвижения управляются одним и тем же слоем управления LSR, куда наряду с традиционными протоколами IP-маршрутизации, такими как RIP, OSPF, IS-IS и BGP, входят и новые протоколы, называемые сигнальными. **Сигнальные протоколы** нужны для автоматического установления в сети виртуального пути, называемого в технологии MPLS **путем коммутации по меткам** (Label Switching Path, **LSP**). Общий слой управления позволяет LSR гибко использовать наличие двух модулей продвижения — одну часть

---

[1] Заметим, что на практике такое устройство чаще всего по-прежнему называют IP-маршрутизатором или IP/MPLS-маршрутизатором.

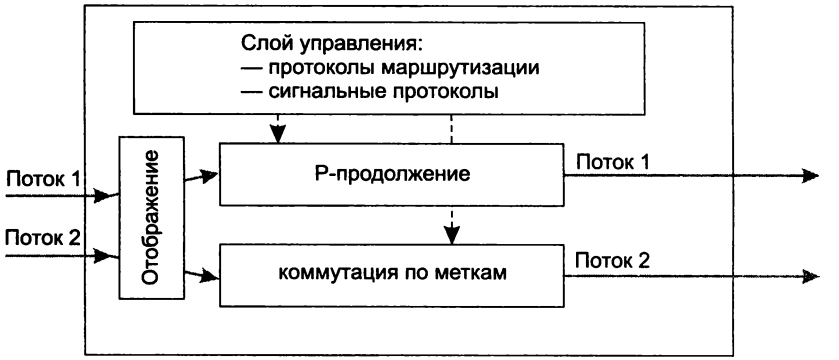


Рис. 20.1. Структура LSR

потоков данных он может продвигать, применяя технику IP-продвижения, а другую — технику коммутации по меткам. Слой управления имеет информацию о топологии сети, необходимую для работы каждого уровня продвижения.

Какие именно потоки нужно продвигать, тем или иным способом решает администратор LSR, который конфигурирует его соответствующим образом. В зависимости от параметров конфигурации IP/MPLS-маршрутизатор выполняет *отображение* входных потоков пакетов на модуль IP-продвижения или на модуль коммутации по меткам. Поток выделяется в соответствии с его признаками, к которым могут относиться IP-адреса, MAC-адреса, порты TCP/UDP и другие поля заголовка пакета и кадра, обычно используемые для классификации потоков данных.

Пути коммутации по меткам прокладываются в сети независимо от того, существует ли поток пакетов в сети в данное время или только является *топологически* возможным. Последнее условие означает, что в сети имеются некоторые два конечных узла, определяемые IP-адресами, и между ними есть возможность установить путь через промежуточные IP/MPLS-маршрутизаторы. Такое свойство путей коммутации по меткам иногда называют «топологически ведомым» (topology-driven).

Здесь нужно отметить, что при разработке технологии MPLS обсуждалась и другая идея — идея установления путей коммутации по меткам в зависимости от характеристик существующих в сети потоков. Такой способ установления путей коммутации по меткам можно было бы назвать «ведомым данными» (data-driven). Как вы помните из сравнения свойств дейтаграммных технологий и технологий виртуальных каналов, серьезным недостатком коммутируемых виртуальных путей является их неэффективность при передаче кратковременных потоков из-за внесения задержки при их установлении. Имея в виду этот недостаток, сторонники техники «ведомых данными» путей коммутации по меткам (MPLS) предлагали динамически создавать такие пути только для долговременных потоков (то есть только после того, как некоторый поток покажет свою «долговременность»), а пакеты кратковременных потоков передавать путем стандартного IP-продвижения, эффективного для этого типа потоков. Однако такое кардинальное изменение логики работы коммутационных устройств, требующее измерения длительности существования потока, было в конце концов отвергнуто и заменено более привычным способом формирования таблиц продвижения заранее (до появления потока) в соответствии с топологией сети.

В заключение этого краткого обзора идеи совмещения техники дейтаграммной передачи и виртуальных каналов хочется еще раз подчеркнуть значение *единого слоя управления* в устройстве LSR. Давайте вернемся к главе 18 и еще раз взглянем на рис. 18.5. На нем показаны два сетевых уровня — уровень IP и уровень канального протокола. Если считать, что на канальном уровне коммутаторы поддерживают технологию MPLS, то чем такая двухслойная структура сети отличается от двухслойной структуры устройства LSR? И почему с помощью двухслойной сети нельзя оказывать те же услуги и с таким же удобством для оператора сети, что и с помощью однослойной внешне, но двухслойной внутренне сети устройств LSR? Основное отличие как раз и заключается в том, что в случае двухслойной сети каждый слой продвижения данных находится под контролем собственного слоя управления. Эти слои управления были созданы без учета существования друг друга в одной сети, поэтому заставить их работать вместе и скоординированно не просто. Скорее всего, администратор сети сможет использовать факт наличия в сети двух слоев, способных предоставлять транспортные услуги, простым физическим подключением различных клиентов ко входным интерфейсам разных слоев.

Наличие единого слоя управления в устройствах LSR как раз и создает возможность координировать работу двух разных слоев продвижения пакетов из одного центра. Протоколы маршрутизации и сигнализации в этом случае явно учитывают существование двух способов продвижения пакетов, при этом как автоматически, так и с учетом конфигурации, заданной администратором, обслуживают потоки данных пользователей наиболее рациональным способом.

Пути коммутации по меткам

Пути коммутации по меткам в технологии MPLS представляют собой некоторый гибрид коммутируемых и постоянных виртуальных каналов. Их можно отнести к коммутируемым, так как они устанавливаются в сети автоматически с помощью сигнальных протоколов. В то же время они могут считаться постоянными, так как их создание инициируется не динамическим запросом, вызванным необходимостью установить сеанс связи между конечными узлами и передать некоторые данные, а имеющейся топологией сети, мало изменяющейся во времени.

Пути коммутации по меткам в технологии MPLS поддерживают инжиниринг трафика за счет соответствующих протоколов маршрутизации и специального сигнального протокола.

Рассмотрим работу пути LSP на примере сети, показанной на рис. 20.2. Эта MPLS-сеть взаимодействует с несколькими IP-сетями, возможно, не поддерживающими технологию MPLS.

На рисунке мы видим новый тип устройств — это пограничные устройства LSR, которые в технологии MPLS имеют специальное название — «пограничные коммутирующие по меткам маршрутизаторы» (Label switch Edge Router, **LER**).

Устройство LER, являясь функционально более сложным, принимает трафик от других сетей в форме стандартных IP-пакетов, а затем добавляет к каждому пакету метку и направляет вдоль соответствующего пути к выходному устройству LER через несколько промежуточных устройств LSR. При этом пакет продвигается не на основе IP-адреса назначения, а на основе метки.

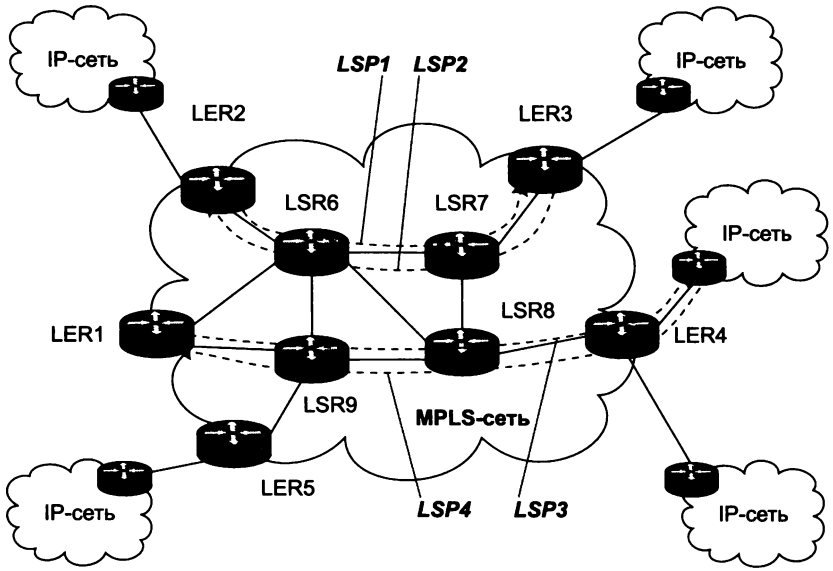


Рис. 20.2. MPLS-сеть

Как и в других технологиях, использующих технику виртуальных каналов, метка имеет локальное значение в пределах каждого устройства LER или LSR, то есть при передаче пакета с входного интерфейса на выходной выполняется смена значения метки.

При принятии решения о выборе следующего хопа блок продвижения по меткам использует *таблицу коммутации*, которая в стандарте MPLS носит название **таблицы продвижения**. Таблица продвижения в технологии MPLS похожа на аналогичные таблицы других технологий, основанных на технике виртуальных каналов (табл. 20.1).

Таблица 20.1. Пример таблицы продвижения в технологии MPLS

Входной интерфейс	Метка	Следующий хоп	Действия
S0	245	S1	256
S0	27	S2	45
...	...	...	...

Внимательный читатель заметил, наверное, небольшое отличие данной таблицы от обобщенной таблицы коммутации, представленной на рис. 19.1 (см. главу 19). Действительно, вместо поля выходного интерфейса здесь поле следующего хопа, а вместо поля выходной метки — поле действий. В большинстве случаев обработки MPLS-кадров эти поля используются точно таким же образом, как соответствующие им поля обобщенной таблицы коммутации. То есть значение поля следующего хопа является значением интерфейса, на который нужно передать кадр, а значение поля действий — новым значением метки. Однако в некоторых случаях эти поля служат другим целям (см. далее).

Рассматриваемые таблицы для каждого устройства LSR формируются *сигнальным протоколом*. В MPLS используется два различных сигнальных протокола: **протокол распределения меток** (Label Distribution Protocol, LDP) и модификация уже знакомого нам протокола резервирования ресурсов RSVP.

Формируя таблицы продвижения на устройствах LSR, сигнальный протокол прокладывает через сеть виртуальные маршруты, которые в технологии MPLS называют **путями коммутации по меткам** (Label Switching Path, LSP).

В том случае, когда метки устанавливаются в таблицах продвижения с помощью протокола LDP, маршруты виртуальных путей LSP совпадают с маршрутами IP-трафика, так как они выбираются обычными протоколами маршрутизации стека TCP/IP. Модификация протокола RSVP, который изначально был разработан для резервирования параметров QoS (см. раздел «Система интегрированного обслуживания» в главе 17), используется для прокладки путей, выбранных в соответствии с техникой инжиниринга трафика, поэтому эта версия протокола получила название RSVP TE (Traffic Engineering). Можно также формировать таблицы MPLS-продвижения вручную, создавая там статические записи, подобные статическим записям таблиц маршрутизации.

LSP представляет собой *однонаправленный* виртуальный канал, поэтому для передачи трафика между двумя устройствами LER нужно установить по крайней мере два пути коммутации по меткам — по одному в каждом направлении. На рис. 20.2 показаны две пары путей коммутации по меткам, соединяющие устройства LER2 и LER3, а также LER1 и LER4.

LER выполняет такую важную функцию, как направление входного трафика в один из исходящих из LER путей LSP. Для реализации этой функции в MPLS введено понятие **класса эквивалентности продвижения** (Forwarding Equivalence Class, FEC).

Класс эквивалентности продвижения — это группа IP-пакетов, имеющих одни и те же требования к условиям транспортировки (транспортному сервису). Все пакеты, принадлежащие к данному классу, продвигаются через MPLS-сеть по одному виртуальному пути LSP.

Входящий пакет относят к тому или иному классу на основании некоторых признаков. Вот несколько примеров классификации:

- *На основании IP-адреса назначения.* Это наиболее близкий к принципам работы IP-сетей подход, который состоит в том, что для каждого префикса сети назначения, имеющегося в таблице LER-маршрутизации, создается отдельный класс FEC. Протокол LDP, который мы далее рассмотрим, полностью автоматизирует процесс создания классов FEC по этому способу.
- *В соответствии с требованиями инжиниринга трафика.* Классы выбираются таким образом, чтобы добиться баланса загрузки каналов сети.
- *В соответствии с требованиями VPN.* Для конкретной виртуальной частной сети клиента создается отдельный класс FEC.
- *По типам приложений.* Например, трафик IP-телефонии (RTP) составляет один класс FEC, а веб-трафик — другой.
- *По интерфейсу, с которого получен пакет.*

- *По MAC-адресу назначения кадра*, если это кадр Ethernet.

Как видно из приведенных примеров, при классификации трафика в MPLS могут использоваться признаки, не только взятые из заголовка IP-пакета, но и многие другие, включая информацию канального (MAC-адрес) и физического (интерфейс) уровней.

После принятия решения о принадлежности пакета к определенному классу FEC его нужно связать с существующим путем LSP. Для этой операции LER использует таблицу FTN (FEC To Next hop — отображение класса FEC на следующий хоп). Таблица 20.2 представляет собой пример FTN.

Таблица 20.2. Пример таблицы FTN

Признаки FEC	Метка
123.20.0.0/16; 195.14.0.0/16	106
194.20.0.0/24; eth1	107

На основании таблицы FTN каждому входящему пакету назначается соответствующая метка, после чего этот пакет становится неотличим в домене MPLS от других пакетов того же класса FEC, все они продвигаются по одному и тому же пути внутри домена.

У администратора сети имеется возможность формировать таблицы FEC или же корректировать их, если они формируются автоматически.

Сложная настройка и конфигурирование выполняются только в LER, а все промежуточные устройства LSR делают простую работу, продвигая пакет в соответствии с техникой виртуального канала.

Выходное устройство LER удаляет метку и передает пакет в следующую сеть уже в стандартной форме IP-пакета. Таким образом, технология MPLS остается прозрачной для остальных IP-сетей.

Обычно в MPLS-сетях используется усовершенствованный по сравнению с описанным алгоритм обработки пакетов. Усовершенствование заключается в том, что удаление метки выполняет не последнее на пути устройство, а *предпоследнее*. Действительно, после того как предпоследнее устройство определит на основе значения метки следующий хоп, метка в MPLS-кадре уже не нужна, так как последнее устройство, то есть выходное устройство LER, должно продвигать пакет на основе значения IP-адреса. Это небольшое изменение алгоритма продвижения кадра позволяет сэкономить одну операцию над MPLS-кадром. В противном случае последнее вдоль пути устройство должно было бы удалить метку, а уже затем выполнить просмотр таблицы IP-маршрутизации. Эта техника получила название техники **удаления метки на предпоследнем хопе** (Penultimate Hop Popping, PHP).

Заголовок MPLS и технологии канального уровня

Заголовок MPLS состоит из нескольких полей (рис. 20.3):

- *Метка* (20 бит). Используется для выбора соответствующего пути коммутации по меткам.

- *Время жизни* (TTL). Данное поле, занимающее 8 бит, дублирует аналогичное поле IP-пакета. Это необходимо для того, чтобы устройства LSR могли отбрасывать «заблудившиеся» пакеты только на основании информации, содержащейся в заголовке MPLS, не обращаясь к заголовку IP.
- *Класс услуги* (Class of Service, CoS). Поле CoS, занимающее 3 бита, первоначально было зарезервировано для развития технологии, но в последнее время используется в основном для указания класса трафика, требующего определенного уровня QoS.
- *Признак дна стека меток*. Этот признак (S) занимает 1 бит.

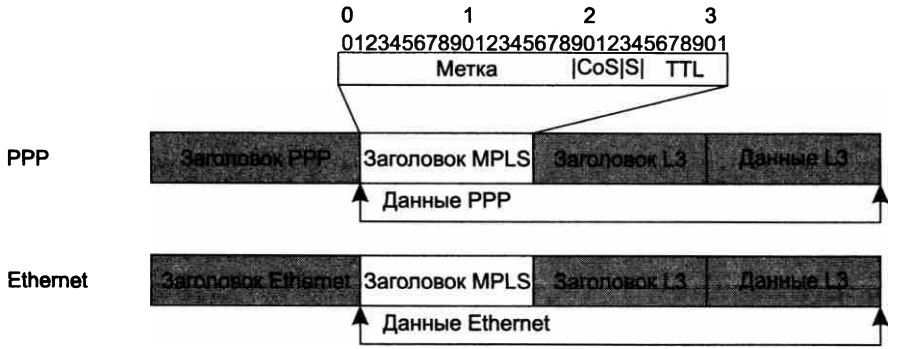


Рис. 20.3. Форматы заголовков нескольких разновидностей технологии MPLS

Концепцию стека меток мы изучим в следующем разделе, а пока для пояснения механизма взаимодействия MPLS с технологиями канального уровня рассмотрим ситуацию, когда заголовок MPLS включает только одну метку.

В кадрах канального уровня заголовок MPLS помещается между оригинальным заголовком и заголовком пакета третьего уровня. На рис. 20.3 этот способ размещения метки показан для кадров PPP и Ethernet. Стандарты MPLS определяют также способ размещения метки в кадрах Frame Relay и ячейках ATM.

В связи с тем, что заголовок MPLS помещается между заголовком канального уровня и заголовком IP, его называют **заголовком-вставкой** (shim header).

Продвижение кадра в MPLS-сети происходит на основе метки MPLS и техники LSP, а не на основе адресной информации и техники той технологии, формат кадра которой MPLS использует. Таким образом, если в MPLS применяется кадр Ethernet, то MAC-адреса источника и приемника, хотя и присутствуют в соответствующих полях кадра Ethernet, для продвижения кадров в соединениях Ethernet с двухточечной топологией не используются. Исключение составляет случай, когда между двумя соседними устройствами LSR находится сеть коммутаторов Ethernet, — тогда MAC-адрес назначения MPLS-кадра потребуется для того, чтобы кадр дошел до следующего устройства LSR, а уже оно будет продвигать его на основании метки. Нахождение MAC-адреса следующего LSR будет в этом случае выполнено стандартным способом с помощью протокола ARP по IP-адресу LSR.

Далее для определенности при рассмотрении примеров мы будем подразумевать, что используется формат кадров MPLS/PPP.

Стек меток

Наличие **стека меток** является одним из оригинальных свойств MPLS.

Стек меток позволяет создавать систему агрегированных путей LSP с любым количеством уровней иерархии. Для поддержания этой функции MPLS-кадр, который перемещается вдоль иерархически организованного пути, должен включать столько заголовков MPLS, сколько уровней иерархии имеет путь. Напомним, что заголовок MPLS каждого уровня имеет собственный набор полей: метка, CoS, TTL и S. Последовательность заголовков организована как стек, так что всегда имеется метка, находящаяся на вершине стека, и метка, находящаяся на дне стека, при этом последняя сопровождается признаком S = 1. Над метками выполняются следующие операции, задаваемые в поле действий таблицы продвижения:

- *Push* — поместить метку в стек. В случае пустого стека эта операция означает простое присвоение метки пакету. Если же в стеке уже имеются метки, в результате этой операции новая метка сдвигает «старые» в глубь стека, сама оказываясь на вершине.
- *Swap* — заменить текущую метку новой.
- *Pop* — выталкивание (удаление) верхней метки, в результате все остальные метки стека поднимаются на один уровень.

Продвижение MPLS-кадра всегда происходит на основе метки, находящейся в данный момент на вершине стека.

Иерархия меток чаще всего находит свое применение в сетях, разделенных на несколько доменов. Внутри домена продвижение пакетов происходит на основе меток одного из уровней стека, а между доменами — на основе меток другого уровня. Такой подход позволяет независимо организовать внутридоменную и междоменную маршрутизацию пакетов, что во многих случаях оказывается полезным. Здесь можно провести аналогию с использованием MAC-адресов для передачи пакетов внутри IP-подсети и IP-адресов для передачи пакетов между IP-подсетями. Стек меток также оказывается полезным при организации сервиса VPN, с соответствующими примерами мы познакомимся в главе 22.

Рассмотрим работу двух уровней иерархии меток на примере сети, изображенной на рис. 20.4.

Сеть состоит из трех MPLS-доменов. На рисунке показаны путь LSP1 в домене 1 и путь LSP2 в домене 2. LSP1 соединяет устройства LER1 и LER2, проходя через устройства LSR1, LSR2 и LSR3. Пусть начальной меткой пути LSP1 является метка 256, которая была присвоена пакету пограничным устройством LER1. На основании этой метки пакет поступает на устройство LSR1, которое по своей таблице продвижения определяет новое значение метки пакета (272) и переправляет его на вход LSR2. Устройство LSR2, действуя аналогично, присваивает пакету новое значение метки (132) и передает его на вход LSR3. Устройство LSR3, будучи предпоследним устройством в пути LSP1, выполняет операцию *Pop* и удаляет метку из стека. Устройство LER2 продвигает пакет уже на основании IP-адреса.

На рисунке также показан путь LSP2 в домене 2. Он соединяет устройства LER3 и LER4, проходя через устройства LSR4, LSR5 и LSR6, и определяется последовательностью меток 188, 112, 101.

Для того чтобы IP-пакеты могли передаваться на основе технологии MPLS не только внутри каждого домена, но и между доменами (например, между устройствами LER1 и LER4), существуют два принципиально разных решения.

- Первое решение состоит в том, что между LER1 и LER4 устанавливается один *одноуровневый* путь коммутации по меткам, соединяющий пути LSP1 и LSP2 (которые в этом случае становятся одним путем). Это простое на первый взгляд решение, называемое **сшиванием** путей LSP, плохо работает в том случае, когда MPLS-домены принадлежат разным поставщикам услуг, не позволяя им действовать независимо друг от друга, так как путь должен быть установлен «из конца в конец» одним из сигнальных протоколов (мы их рассмотрим далее, сейчас детали их работы нам не важны).
- Вторым, более перспективным решением является применение *многоуровневого* подхода к соединению двух MPLS-доменов, принадлежащих, возможно, разным поставщикам услуг.

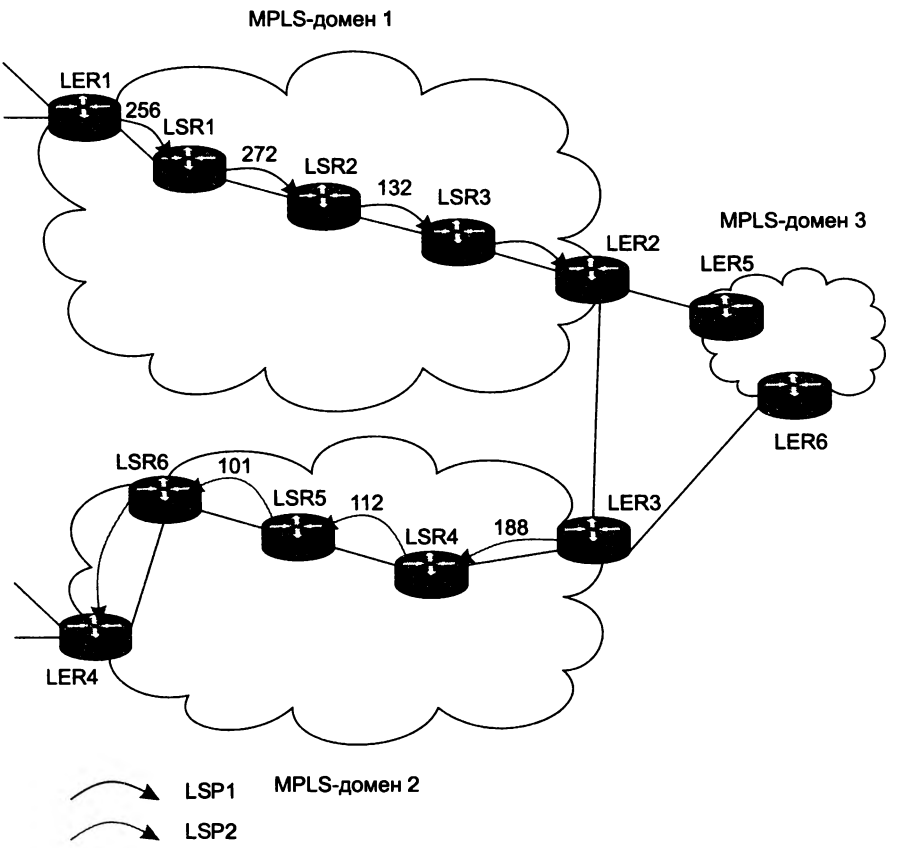


Рис. 20.4. Пути LSP1 и LSP2, проложенные в доменах 1 и 2 MPLS-сети

Для реализации второго подхода в нашем примере нужно создать путь коммутации по меткам второго уровня (LSP3), соединяющий устройства LER1 и LER4. Этот путь определяет последовательность хопов *между доменами*, а не между внутренними устройствами LSR каждого домена. Так, LSP3 состоит из хопов LER1-LER2-LER3-LSR4. В этом отношении многоуровневый подход MPLS концептуально очень близок подходу протокола BGP, определяющего путь между автономными системами.

Рассмотрим более детально, как работает технология MPLS в случае путей коммутации по меткам двух уровней (рис. 20.5).

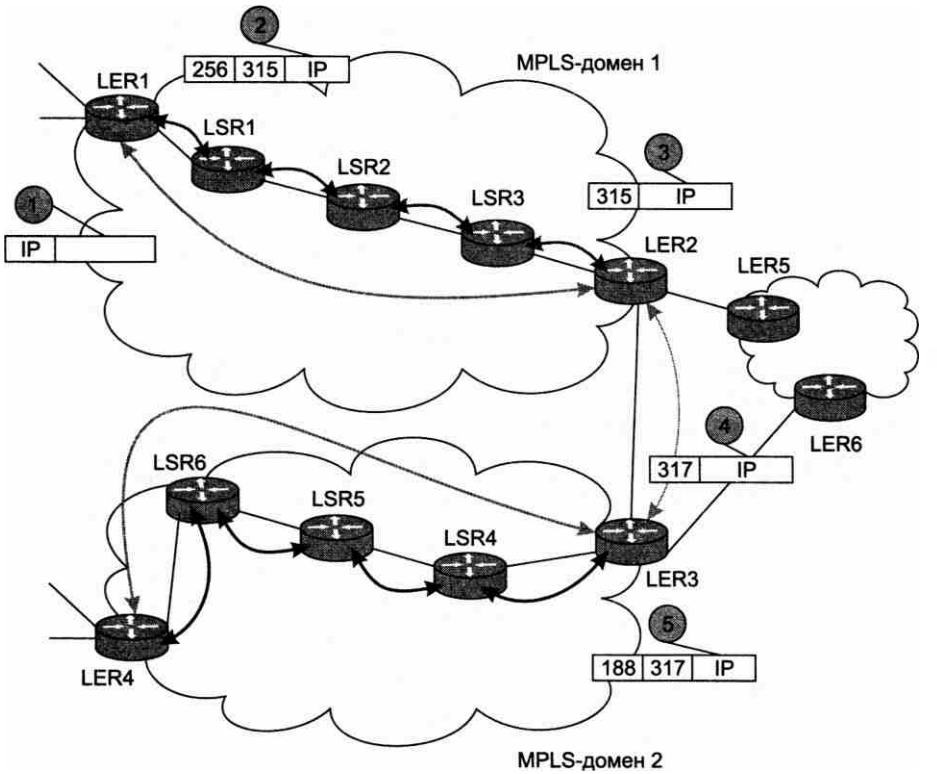


Рис. 20.5. Использование стека меток в иерархии путей

В устройстве LER1 начинаются два пути — LSP1 и LSP3 (последний показан на рисунке серым цветом), что обеспечивается соответствующей записью в таблице продвижения устройства LER1 (табл. 20.3).

Таблица 20.3. Запись в таблице продвижения LER1

Входной интерфейс	Метка	Следующий хоп	Действия
...	...	...	...
S0	—	S1	315 Push 256
...	...	...	...

IP-пакеты, поступающие на интерфейс S0 устройства LER1, продвигаются на его выходной интерфейс S1, где для них создается заголовок MPLS, включающий метку 315 верхнего уровня (LSP3), которая на этот момент является верхушкой стека меток. Затем эта метка проталкивается на дно стека (операция *Push*), а верхней становится метка 256, относящаяся к LSP1.

Далее MPLS-кадр с меткой 256 поступает на выходной интерфейс S1 пограничного устройства LER1 и передается на вход LSR1. Устройство LSR1 обрабатывает кадр в соответствии со своей таблицей продвижения (табл. 20.4). Метка 256, находящаяся на вершине стека, заменяется меткой 272. (Отметьте, что метка 315, находящаяся ниже в стеке, устройством LSR1 игнорируется.)

Таблица 20.4. Запись в таблице продвижения LSR1

Входной интерфейс	Метка	Следующий хоп	Действия
...	...	...	...
S0	256	S1	272
...	...	...	...

Аналогичные действия выполняет устройство LSR2, которое заменяет метку меткой 132 и отправляет кадр следующему по пути устройству LSR3 (табл. 20.5).

Таблица 20.5. Запись в таблице продвижения LSR3

Входной интерфейс	Метка	Следующий хоп	Действия
...	...	...	...
S0	132	S1	Pop
...	...	...	...

Работа устройства LSR3 несколько отличается от работы устройств LSR1 и LSR2, так как оно является *предпоследним* устройством LSR для пути LSP1. В соответствии с записью в табл. 22.4 устройство LSR3 выполняет выталкивание (*Pop*) из стека метки 132, относящейся к пути LSP1, выполняя операцию PHP. В результате верхней меткой стека становится метка 315, принадлежащая пути LSP3.

Устройство LER2 продвигает поступивший на его входной интерфейс S0 кадр на основе своей записи таблицы продвижения (табл. 20.6). Устройство LER3 сначала заменяет метку 315 пути LSP3 значением 317, затем проталкивает ее на дно стека и помещает на вершину стека метку 188, которая является меткой пути LSP2, внутреннего для домена 2. Перемещение кадра вдоль пути LSP2 происходит аналогичным образом.

Таблица 20.6. Запись в таблице продвижения LER3

Входной интерфейс	Метка	Следующий хоп	Действия
...	...	...	...
S0	315	S1	317 Push 188
...	...	...	...

Нужно подчеркнуть, что значение метки междоменного пути LSP3 на границе между доменами не зависит от значений меток, используемых для внутридоменных путей LSP1 и LSP2. Это позволяет операторам доменов изменять значения меток внутридоменных

путей независимо друг от друга, например прокладывая внутридоменные пути по другим маршрутам (а это неизбежно приведет к переназначению меток в каждом из устройств LSR и LER). Важно, что при этом значение междоменной метки при передаче пакета между устройствами LER доменов не меняется, поэтому пакет правильно обрабатывается принимающим устройством LER. Например, LER3 получит пакет от LER6 со значением метки 317 независимо от того, какое значение имела метка внутридоменного пути LSP1. При «сшивании» одноуровневых устройств LSP такой независимости доменов добиться нельзя.

Описанная модель двухуровневого пути легко может быть расширена для любого количества уровней.

Протокол LDP

Протокол распределения меток (Label Distribution Protocol, LDP) позволяет автоматически создавать в сети пути LSP в соответствии с *существующими* в таблицах маршрутизации записями о маршрутах в IP-сети. Протокол LDP является сигнальным протоколом сетей MPLS.

Протокол LDP принимает во внимание только те записи таблицы маршрутизации, которые созданы с помощью внутренних протоколов маршрутизации, то есть протоколов типа IGP, поэтому режим автоматического создания LSP с помощью протокола LDP иногда называют режимом MPLS IGP (в отличие от режима MPLS TE, когда маршруты выбираются из соображений инжиниринга трафика и не совпадают с маршрутами, выбранными внутренними протоколами маршрутизации). Спецификация LDP дана в документе RFC 5036.

Рассмотрим работу протокола LDP на примере сети, изображенной на рис. 20.6.

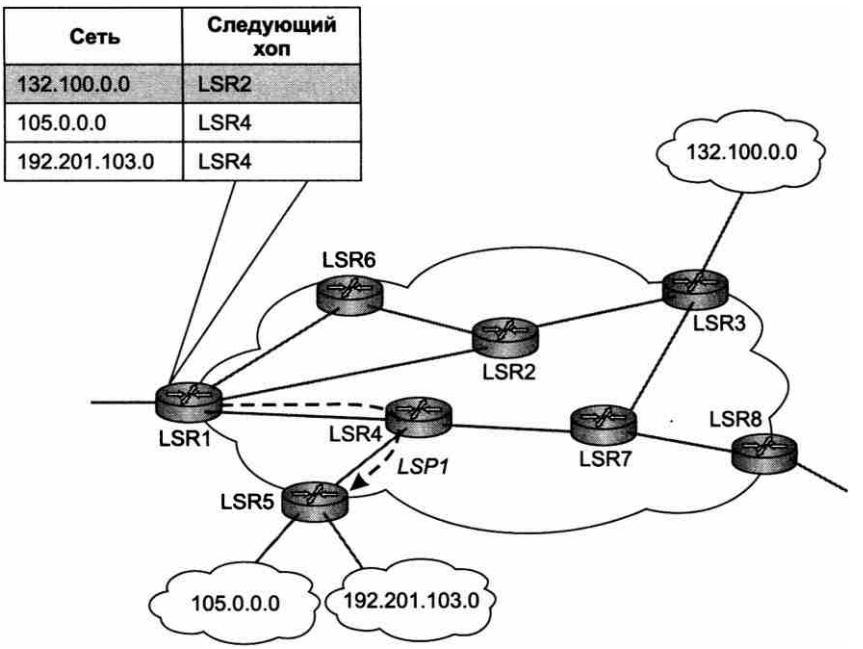


Рис. 20.6. MPLS-сеть с устройствами LSR, поддерживающими LDP

Все устройства LSR сети поддерживают сигнальный протокол LDP. От устройства LSR1 в сети уже установлен один путь LSP1 — по этому пути идет трафик к сетям 105.0.0.0 и 192.201.103.0. Это означает, что таблица FTN (отображающая сети назначения на устройства LSP) у LSR1 соответствует табл. 20.7.

Таблица 20.7. Таблица FTN устройства LSR1

Признаки FEC	Метка
105.0.0.0; 192.201.103.0	231

Метка 231 в этой таблице соответствует пути LSP1.

Мы рассмотрим функционирование протокола LDP в ситуации, когда в результате работы протоколов маршрутизации или же после ручной модификации администратором сети в таблице маршрутизации устройства LSR1 появилась запись о новой сети назначения, для которой в сети поставщика услуг еще не проложен путь коммутации по меткам. В нашем случае это сеть 132.100.0.0 (она закрашена в таблице маршрутизации LSR1) и для нее нет записи в таблице FTN.

В этом случае устройство LSR1 автоматически инициирует процедуру прокладки нового пути. Для этого оно запрашивает по протоколу LDP метку для новой сети 132.100.0.0 у маршрутизатора, IP-адрес которого в таблице маршрутизации указан для данной сети как адрес следующего хопа.

Однако для того чтобы воспользоваться протоколом LDP, нужно сначала установить между устройствами LSR сеанс LDP, так как этот протокол работает в режиме установления соединений.

Сеансы LDP устанавливаются между соседними маршрутизаторами автоматически. Для этого каждое устройство LSR, на котором развернут протокол LDP, начинает посылать своим соседям сообщения *Hello*. Эти сообщения посылаются по групповому IP-адресу 224.0.0.2, который является адресом всех маршрутизаторов подсети. Если соседний маршрутизатор также поддерживает протокол LDP, то он в ответ устанавливает сеанс TCP через порт 646 (этот порт закреплен за протоколом LDP).

В результате обмена сообщениями *Hello* все поддерживающие протокол LDP устройства LSR обнаруживают своих соседей и устанавливают с ними сеансы, как показано на рис. 20.7 (для простоты на рисунке представлены не все сеансы LDP, существующие в сети).

Будем считать, что между устройствами LSR1 и LSR2 установлен сеанс LDP.

Тогда при обнаружении новой записи в таблице маршрутизации, указывающей на устройство LSR2 в качестве следующего хопа, устройство LSR1 просит устройство LSR2 назначить метку для нового пути к сети 132.100.0.0. Говорят, что устройство LSR2 находится ниже по потоку (downstream) относительно устройства LSR1 на пути к сети 132.100.0.0. Соответственно устройство LSR1 расположено выше по потоку для устройства LSR2 относительно сети 132.100.0.0. Естественно, что для других сетей назначения у устройства LSR1 имеются другие соседи вниз по потоку, а у устройства LSR2 — другие соседи вверх по потоку.

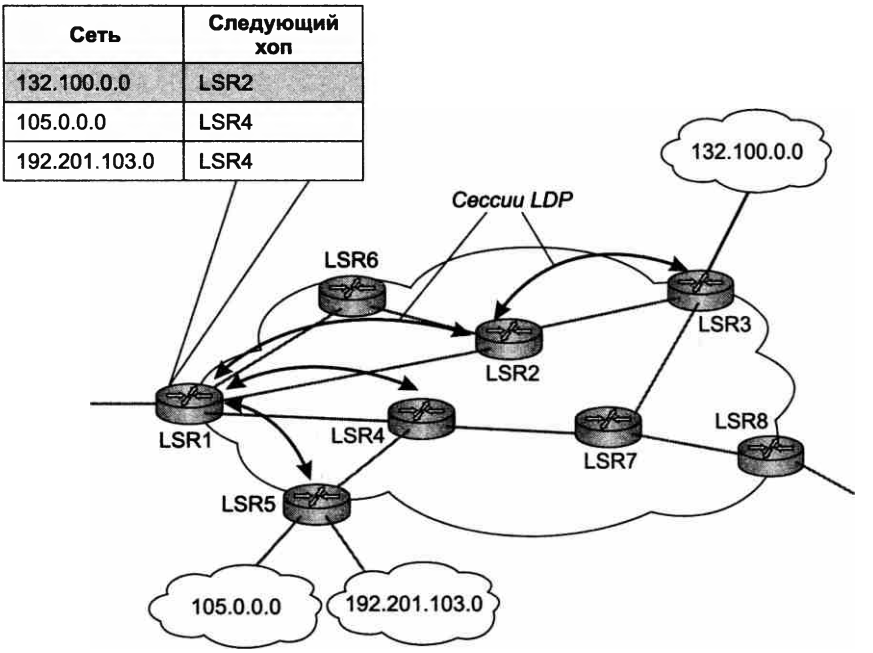


Рис. 20.7. Сеансы LDP устанавливаются между непосредственными соседями

Причина, по которой значение метки для нового пути выбирается соседом ниже по потоку, понятна — эта метка, которая имеет локальное значение на двухточечном соединении между соседними устройствами, будет использоваться именно этим устройством для того, чтобы понимать, к какому пути LSP относится пришедший MPLS-кадр. Поэтому устройство ниже по потоку выбирает уникальное значение метки, исходя из неиспользованных значений меток для своего интерфейса, который связывает его с соседом выше по потоку.

Для получения значения метки устройство LSR1 выполняет запрос метки протокола LDP. Формат такого запроса достаточно прост (рис. 20.8).

Запрос метки (0×0401)	Длина сообщения
Идентификатор сообщения	
Элемент FEC	

Рис. 20.8. Формат LDP-запроса метки

Идентификатор сообщения требуется для того, чтобы при получении ответа можно было однозначно сопоставить ответ некоторому запросу (устройство может послать несколько запросов до получения ответов на каждый из них).

В нашем примере в качестве элемента FEC указан адрес 132.100.0.0.

Устройство LSR2, приняв запрос, находит, что у него также нет проложенного пути к сети 132.100.0.0, поэтому оно передает LDP-запрос следующему устройству LSR, адрес которого указан в его таблице маршрутизации в качестве следующего хопа для сети 132.100.0.0. В примере, показанном на рис. 20.7, таким устройством является LSR3, на котором путь коммутации по меткам должен закончиться, так как следующий хоп ведет за пределы MPLS-сети данного оператора.

ПРИМЕЧАНИЕ

Возникает вопрос: как устройство LSR3 узнает о том, что является последним в сети поставщика услуг на пути к сети 132.100.0.0? Дело в том, что сеансы LDP устанавливаются только между устройствами одного поставщика услуг, поэтому отсутствие сеанса LDP со следующим хопом маршрута и говорит устройству LSR, что оно является последним в своем домене для данного пути LSP.

Устройство LSR3, обнаружив, что для пути к сети 132.100.0.0 оно является пограничным, назначает для прокладываемого пути метку, еще не занятую его входным интерфейсом S0, и сообщает об этой метке устройству LSR2 в LDP-сообщении, формат которого представлен на рис. 20.9. Пусть это будет метка 231.

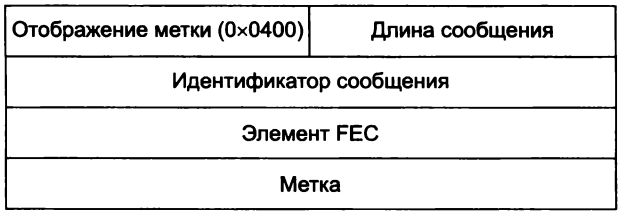


Рис. 20.9. Формат отображения метки на элемент FEC протокола LDP

В свою очередь устройство LSR2 назначает не используемую его интерфейсом S0 метку и сообщает об этом в LDP-сообщении отображения метки устройству LSR1. После этого новый путь коммутации по меткам, ведущий от LSR1 к сети 132.100.0.0, считается проложенным (рис. 20.10), и вдоль него пакеты начинают передаваться уже на основе меток и таблиц продвижения, а не IP-адресов и таблиц маршрутизации.

Было бы нерационально прокладывать отдельный путь для каждой сети назначения каждого маршрутизатора. Поэтому устройства LSR стараются строить агрегированные пути коммутации по меткам и передавать вдоль них пакеты, следующие к некоторому набору сетей. Так, на рис. 20.10 устройство LSR1 передает по пути LSP2 пакеты, следующие не только к сети 132.100.0.0, но и к сетям 194.15.17.0 и 201.25.10.0, информация о которых появилась уже после того, как путь LSP2 был проложен.

Мы рассмотрели только один режим работы протокола LDP, который носит длинное название «Упорядоченный режим управления распределением меток с запросом устройства вниз по потоку». Здесь под упорядоченным режимом понимается такой режим, когда некоторое промежуточное устройство LSR не передает метку для нового пути устройству LSR, лежащему выше по потоку, до тех пор пока не получит метку для этого пути от устройства LSR, лежащего ниже по потоку. В нашем случае устройство LSR2 ждало получения метки от LSR3 и уже потом передало метку устройству LSR1.

Сеть	Следующий хоп
132.100.0.0	LSR2
194.15.17.0	LSR2
201.25.10.0	LSR2
105.0.0.0	LSR4
192.201.103.0	LSR4

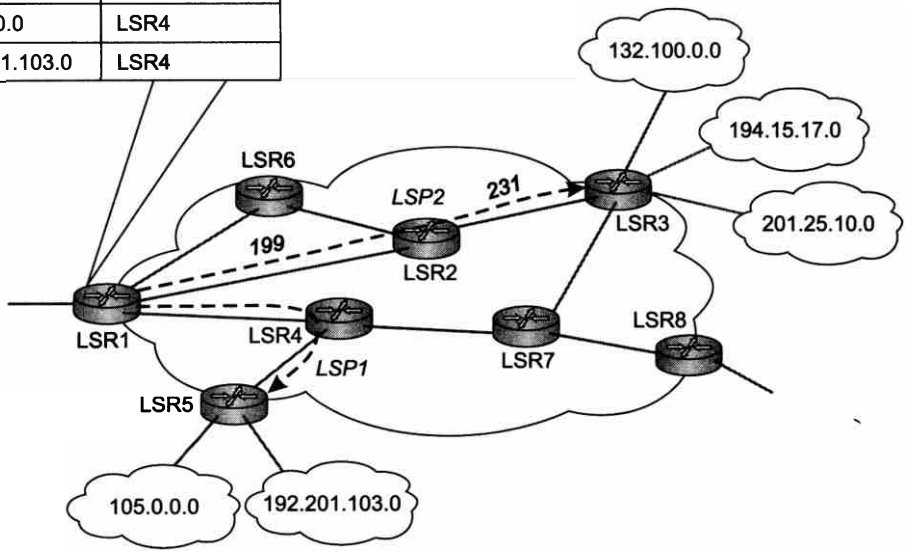


Рис. 20.10. Новый путь LSP2

Существует и другой режим управления распределением меток, который называется независимым. При независимом управлении распределением меток LSR может назначить и передать метку, не дожидаясь прихода сообщения от своего соседа, лежащего ниже по потоку. Например, устройство LSR2 могло бы назначить и передать метку 199 устройству LSR1, не дожидаясь прихода метки 231 от устройства LSR3. Так как метки имеют локальное значение, результат изменения режима остался бы прежним.

Существуют также два метода распределения меток: распределение по запросу от лежащего ниже по потоку устройства и без запроса. Для нашего случая это означает, что если бы устройство LSR2 обнаружило в своей таблице маршрутизации запись о новой сети 132.100.0.0, оно могло бы назначить метку новому пути и передать ее устройству LSR1 без запроса. Так как при этом устройство LSR2 не знает своего соседа выше по потоку (таблица маршрутизации не говорит об этом), оно передает эту информацию всем своим соседям по сеансам LDP. В этом варианте работы протокола LDP устройства LSR могут получать альтернативные метки для пути к некоторой сети; а выбор наилучшего пути осуществляется обычным для IP-маршрутизаторов (которыми устройства LSR являются по совместительству) способом — на основании наилучшей метрики, выбираемой протоколом маршрутизации.

Как видно из описания, существует два независимых параметра, которые определяют вариант работы протокола LDP: режим управления распределением меток и метод рас-

пределения меток. Так как каждый параметр имеет два значения, всего существует четыре режима работы протокола LDP.

Протокол LDP чаще всего функционирует в режиме независимого управления распределением меток без запроса.

Упорядоченное управление распределением меток требуется при прокладке путей LSP, необходимых для инжиниринга трафика.

Инжиниринг трафика в MPLS

Технология MPLS поддерживает технику инжиниринга трафика, описанную в главе 6. В этом случае используются модифицированные протоколы сигнализации и маршрутизации, имеющие приставку TE (Traffic Engineering — инжиниринг трафика). В целом такой вариант MPLS получил название MPLS TE.

В технологии MPLS TE пути LSP называют **TE-туннелями**. TE-туннели не прокладываются распределенным способом вдоль путей, находимых обычными протоколами маршрутизации независимо в каждом отдельном устройстве LSR. Вместо этого TE-туннели прокладываются в соответствии с техникой маршрутизации от источника, когда централизованно задаются промежуточные узлы маршрута. В этом отношении TE-туннели подобны постоянным виртуальным каналам технологий ATM и Frame Relay. Инициатором задания маршрута для TE-туннеля выступает начальный узел туннеля, а рассчитываться такой маршрут может как этим же начальным узлом, так и внешней по отношению к сети программной системой или администратором.

MPLS TE поддерживает туннели двух типов:

- **строгий TE-туннель** определяет все промежуточные узлы между двумя пограничными устройствами;
- **свободный TE-туннель** определяет только часть промежуточных узлов от одного пограничного устройства до другого, а остальные промежуточные узлы выбираются устройством LSR самостоятельно.

На рис. 20.11 показаны оба типа туннелей.

Туннель 1 является примером строгого туннеля, при его задании внешняя система (или администратор сети) указала как начальный и конечный узлы туннеля, так и все промежуточные узлы, то есть последовательность IP-адресов для устройств LER1, LSR1, LSR2, LSR3, LSR4, LER3. Таким образом, внешняя система решила задачу инжиниринга трафика, выбрав путь с достаточной неиспользуемой пропускной способностью. При установлении туннеля 1 задается не только последовательность LSR, но и требуемая пропускная способность пути. Несмотря на то что выбор пути происходит в автономном режиме, все устройства сети вдоль туннеля 1 проверяют, действительно ли они обладают запрошенной неиспользуемой пропускной способностью, и только в случае положительного ответа туннель прокладывается.

При прокладке туннеля 2 (свободного) администратор задает только начальный и конечный узлы туннеля, то есть устройства LER5 и LER2. Промежуточные устройства LSR4 и LSR2 находятся автоматически начальным узлом туннеля 2, то есть устройством LER5, а затем с помощью сигнального протокола устройство LER5 сообщает этим и конечному устройствам о необходимости прокладки туннеля.

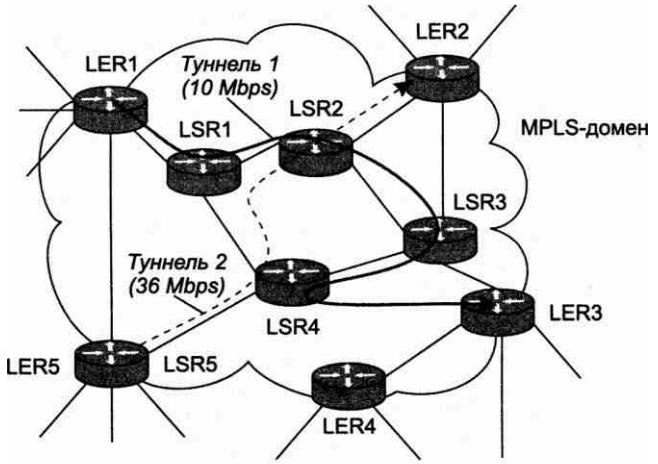


Рис. 20.11. Два типа TE-туннелей в технологии MPLS

Независимо от типа туннеля он всегда обладает таким параметром, как резервируемая пропускная способность. В нашем примере туннель 1 резервирует для трафика 10 Мбит/с, а туннель 2 — 36 Мбит/с. Эти значения определяются администратором, и технология MPLS TE никак не влияет на их выбор, она только реализует запрошенное резервирование. Чаще всего администратор оценивает резервируемую для туннеля пропускную способность на основании измерений трафика в сети, тенденций изменения трафика, а также собственной интуиции. Некоторые реализации MPLS TE позволяют затем автоматически корректировать величину зарезервированной пропускной способности на основании автоматических измерений реальной интенсивности трафика, проходящего через туннель.

Однако сама по себе прокладка в MPLS-сети TE-туннеля еще не означает передачи по нему трафика. Она означает только то, что в сети действительно существует возможность передачи трафика по туннелю со средней скоростью, не превышающей зарезервированное значение. Для того чтобы данные были переданы по туннелю, администратору предстоит еще одна ручная процедура — задание для начального устройства туннеля условий, определяющих, какие именно пакеты должны передаваться по туннелю. Условия могут быть чрезвычайно разнообразными, так, в качестве признаков агрегированного потока, который должен передаваться по туннелю, могут выступать все традиционные признаки: IP-адрес назначения и источника, тип протокола, номера TCP- и UDP-портов, номер интерфейса входящего трафика, значения приоритета в протоколах DSCP и IP и т. д.

Таким образом, устройство LER должно сначала провести *классификацию трафика*, затем выполнить *профилирование*, удостоверившись, что средняя скорость потока не превышает зарезервированную, и, наконец, начать *маркировать* пакеты, используя начальную метку TE-туннеля, чтобы передавать трафик через сеть с помощью техники MPLS. В этом случае расчеты, выполненные на этапе выбора пути для туннеля, дадут нужный результат — баланс ресурсов сети при соблюдении средней скорости для каждого потока.

Однако мы еще не рассмотрели специфический набор протоколов, которые устройства LER и LSR сети используют для прокладки свободных туннелей или проверки работоспособности созданных администратором строгих туннелей.

Для выбора и проверки путей через туннели в технологии MPLS TE используются расширения протоколов маршрутизации, работающих на основе алгоритма состояния связей. Сегодня такие расширения стандартизованы для протоколов OSPF и IS-IS. Для решения задачи TE в протоколы OSPF и IS-IS включены новые типы объявлений, обеспечивающие распространение по сети информации о номинальной и незарезервированной (доступной для TE-потоков) величинах пропускной способности каждой связи. Таким образом, ребра результирующего графа сети, создаваемого в топологической базе каждого устройства LER или LSR, маркируются этими двумя дополнительными параметрами. Располагая таким графом, а также параметрами потоков, для которых нужно определить TE-пути, устройство LER может найти рациональное решение, удовлетворяющее одному из сформулированных в главе 6 ограничений на использование ресурсов сети. Чаще всего решение ищется по наиболее простому критерию, который состоит в минимизации максимального значения коэффициента использования вдоль выбранного пути, то есть критерием оптимизации пути является значение min (max K_i) для всех возможных путей.

В общем случае администратору необходимо проложить несколько туннелей для различных агрегированных потоков. С целью упрощения задачи оптимизации выбор путей для этих туннелей обычно осуществляется по очереди, причем администратор определяет очередность на основе своей интуиции. Очевидно, что поиск TE-путей по очереди снижает качество решения: при одновременном рассмотрении всех потоков в принципе можно было бы добиваться более рациональной загрузки ресурсов.

Пример

В примере, показанном на рис. 20.12, ограничением является максимально допустимое значение коэффициента использования ресурсов, равное 0,65. В варианте 1 решение было найдено при очередности рассмотрения потоков 1, 2, 3. Для первого потока был выбран путь *A-B-C*, так как в этом случае он, с одной стороны, удовлетворяет ограничению (все ресурсы вдоль пути — каналы *A-B*, *A-C* и соответствующие интерфейсы маршрутизаторов — оказываются

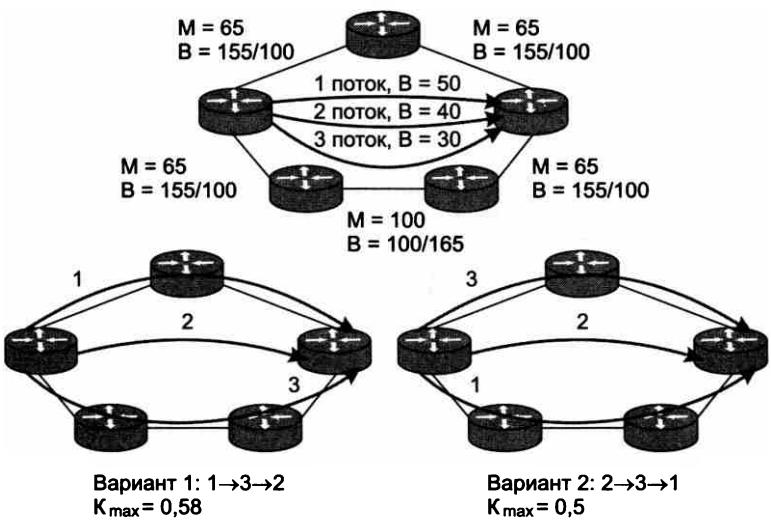


Рис. 20.12. Зависимость качества решения задачи TE от очередности выбора туннелей

загруженными на 50/155 = 0,32), а с другой — обладает минимальной метрикой (65 + 65 = = 130). Для второго потока также был выбран путь *A-B-C*, так как и в этом случае ограничение удовлетворяется: результирующий коэффициент использования оказывается равным 50 + 40/155 = 0,58. Третий поток направляется по пути *A-D-E-C* и загружает ресурсы каналов *A-D*, *D-E* и *E-C* на 0,3. Решение 1 можно назвать удовлетворительным, так как коэффициент использования любого ресурса в сети не превышает 0,58.

Однако существует лучший способ, представленный в варианте 2. Здесь потоки 2 и 3 были направлены по верхнему пути *A-B-C*, а поток 1 — по нижнему пути *A-D-E-C*. Ресурсы верхнего пути оказываются загруженными на 0,45, а нижнего — на 0,5, то есть налицо более равномерная загрузка ресурсов, а максимальный коэффициент использования всех ресурсов сети не превышает 0,5. Этот вариант может быть получен при одновременном рассмотрении всех трех потоков с учетом ограничения min (max *Ki*) или же при рассмотрении потоков по очереди в последовательности 2, 3, 1.

Несмотря на неоптимальность качества решения, в производимом сегодня оборудовании применяется вариант технологии MPLS TE с последовательным рассмотрением потоков. Он проще в реализации и ближе к стандартным для протоколов OSPF и IS-IS процедурам нахождения кратчайшего пути для одной сети назначения (в отсутствие ограничений найденное решение для набора кратчайших путей не зависит от последовательности учета сетей, для которых производился поиск). Кроме того, при изменении ситуации — появлении новых потоков или изменении интенсивности существующих — найти путь удается только для одного потока.

Возможен также подход, в котором внешняя по отношению к сети вычислительная система, работающая в автономном режиме, определяет оптимальное решение для набора потоков. Это может быть достаточно сложная система, которая включает подсистему имитационного моделирования, способную учесть не только средние интенсивности потоков, но и их пульсации и оценить не только загрузку ресурсов, но и результирующие параметры QoS — задержки, потери и т. п. После нахождения оптимального решения его можно модифицировать уже в оперативном режиме поочередного поиска путей.

В технологии MPLS TE информация о найденном рациональном пути используется полностью — то есть запоминаются IP-адреса источника, всех транзитных маршрутизаторов и конечного узла. Поэтому достаточно, чтобы поиском путей занимались только пограничные устройства сети (LER), а промежуточные устройства (LSR) лишь поставляли им информацию о текущем состоянии резервирования пропускной способности каналов.

После нахождения пути независимо от того, найден он был устройством LER или администратором, его необходимо зафиксировать. Для этого в MPLS TE используется расширение уже рассмотренного нами в главе 17 протокола резервирования ресурсов (RSVP), который часто в этом случае называют протоколом **RSVP TE**. Сообщения RSVP TE передаются от одного устройства LSR другому в соответствии с данными о найденных IP-адресах маршрута. При установлении нового пути в сигнальном сообщении наряду с последовательностью адресов пути указывается также резервируемая пропускная способность. Каждое устройство LSR, получив такое сообщение, вычитает запрашиваемую пропускную способность из пула свободной пропускной способности соответствующего интерфейса, а затем объявляет остаток в сообщениях протокола маршрутизации, например CSPF.

В заключение рассмотрим вопрос отношения технологий MPLS TE и QoS. Как видно из описания, основной задачей MPLS TE является использование возможностей технологии MPLS для достижения внутренней цели поставщика услуг, а именно сбалансированной загрузки всех ресурсов своей сети. Однако при этом также создается основа для предоставления транспортных услуг с гарантированными параметрами QoS, так как трафик по TE-туннелям передается при соблюдении некоторого максимального коэффициента использования ресурсов. Как мы знаем из материала главы 7, коэффициент использования ресурсов оказывает решающее влияние на процесс образования очереди, так что потоки, передаваемые по TE-туннелям, передаются с некоторым гарантированным уровнем QoS.

Для того чтобы обеспечить разные параметры QoS для разных классов трафика, поставщику услуг необходимо для каждого класса трафика установить в сети отдельную систему туннелей. При этом для классов чувствительного к задержкам трафика требуется выполнить резервирование таким образом, чтобы максимальный коэффициент использования ресурсов туннеля находился в диапазоне 0,2–0,3, иначе задержки пакетов и их вариации выйдут за допустимые пределы.

Мониторинг состояния путей LSP

Наличие встроенных в транспортную технологию средств мониторинга состояния соединений и локализации ошибок (то есть средств OAM) является необходимым условием для того, чтобы она претендовала на статус технологии операторского класса. В противном случае ее трудно будет использовать операторам сетей, которым нужно обеспечивать своих многочисленных клиентов транспортным сервисом с высоким коэффициентом готовности (в пределах 0,999–0,99999), как это принято в телекоммуникационных сетях.

Первоначально технология MPLS не имела подобных встроенных средств, полагаясь на такие средства стека TCP/IP, как утилиты ping и traceroute, использующие ICMP-сообщения *Echo Request* (эхо-запрос) и *Echo Reply* (эхо-ответ)). Однако классические утилиты ping и traceroute стека TCP/IP не дают корректной информации о состоянии путей LSP, поскольку они могут переноситься как вдоль, так и в обход этих путей с помощью обычной техники продвижения пакетов протокола IP. Поэтому позднее был разработан специальный протокол LSP Ping, который позволяет тестировать работоспособность LSP (режим *ping*) и локализовывать отказы (режим *traceroute*).

Кроме того, для мониторинга состояния LSP можно применять более экономичный, чем LSP Ping, протокол двунаправленного обнаружения ошибок продвижения (см. далее).

Тестирование путей LSP

В протоколе LSP Ping для тестирования состояния LSP применяется техника, близкая к механизму работы утилиты ping протокола IP. Она заключается в том, что протокол LSP Ping отправляет вдоль тестируемого пути LSP сообщение *Echo Request*. Если такое сообщение доходит до устройства LER, которое является конечным узлом тестируемого пути LSP, оно отвечает сообщением *Echo Reply*. Получение исходным узлом такого сообщения означает, что путь LSP работоспособен.

Описанная схема работы аналогична схеме работы утилиты ping протокола IP, однако имеет свои особенности, которые мы поясним на примере сети, изображенной на рис. 20.13.

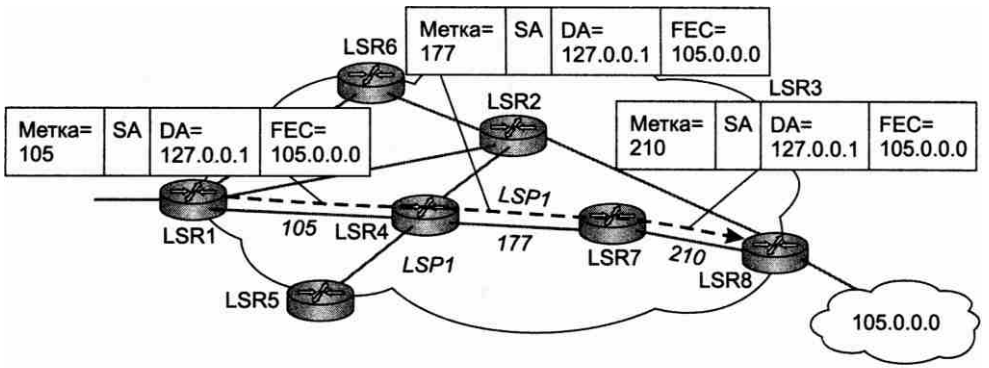


Рис. 20.13. Тестирование LSP с помощью протокола LSP Ping

В этом примере устройство LSR1 тестирует состояние пути LSP1, который заканчивается на устройстве LSR8 (для этого пути оно является устройством LER).

Для тестирования пути LSP1 устройство LSR1 отправляет MPLS-пакет с меткой 105 — эта метка соответствует пути LSP1 на линии между устройствами LSR1 и LSR4. Сообщение *Echo Request* вкладывается в UDP-сообщение, которое в свою очередь вкладывается в IP-пакет. На рисунке показаны только значимые для изучения протокола LSP Ping поля: метка MPLS-кадра, IP-адрес источника (SA), IP-адрес назначения (DA), а также поле FEC, которое идентифицирует тестируемый путь LSP. В нашем примере это IP-адрес сети 105.0.0.0, к которой ведет путь LSP1.

Адрес назначения в IP-пакете, который переносит сообщение *Echo Request*, равен 127.0.0.1, то есть является адресом обратной петли стека протоколов IP каждого узла. О причине использования такого необычного адреса назначения (а не, скажем, IP-адреса интерфейса конечного узла тестируемого пути LSP) мы расскажем позже, а пока заметим, что адрес 127.0.0.1 должен работать правильно, так как в процессе передачи запроса по сети для его продвижения используются MPLS-метки, а не IP-адрес назначения. При приходе на конечный узел IP-пакет освобождается от заголовка MPLS (это также может произойти на предыдущем хопе, если применяется техника PHP) и обрабатывается на основе IP-адреса. Так как адрес 127.0.0.1 указывает на собственный узел, то пакет передается собственному стеку TCP/IP, где он распознается как UDP-пакет протокола LSP Ping и обрабатывается соответственно.

Поле FEC посылается в запросе *Echo Request* для того, чтобы конечный узел пути мог сравнить указанное в пакете значение FEC со значением из его собственной базы данных для пути, по которому пришел кадр запроса. Такой механизм позволяет отслеживать ситуации, когда запрос вследствие каких-то ошибок приходит не по тому пути, который тестируется.

В том случае, когда запрос благополучно доходит до конечного узла пути и тот убеждается, что полученный запрос пришел по нужному пути (то есть полученное значение FEC совпадает со значением FEC из базы данных конечного узла), он отправляет ответ *Echo Reply* узлу, выполнившему запрос. В нашем случае узел LSR8 отправляет ответ *Echo Reply* узлу LSR1. Сообщение *Echo Reply* посылается уже не по пути LSP, а как обычное UDP-сообщение, вложенное в IP-пакет. Если вспомнить, что пути LSP являются одно-

направленными, станет понятно, что это единственное гарантированное решение, так как обратного пути от LSR8 к LSR1 может не существовать.

Теперь посмотрим, что происходит в том случае, когда по какой-то причине путь LSP поврежден. На рис. 20.14 представлен именно такой случай, когда путь поврежден на последнем своем участке (между устройствами LSR7 и LSR8).

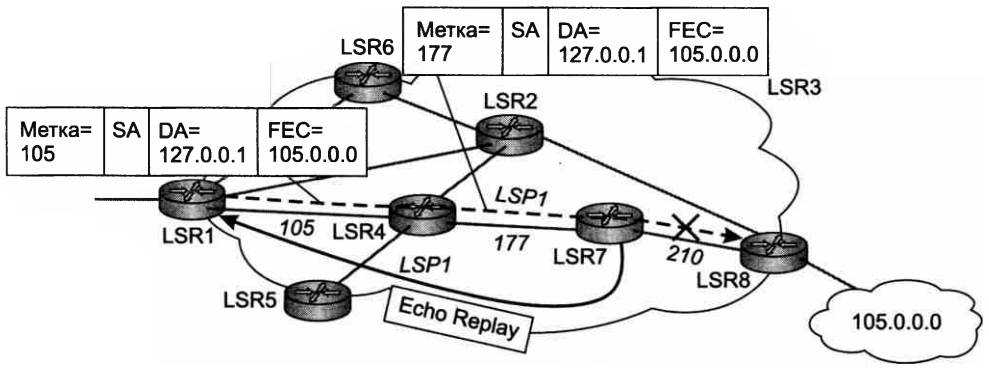


Рис. 20.14. Тестирование неисправного пути LSP с помощью протокола LSP Ping

В этой ситуации LSR7 не может отправить MPLS-кадр по назначению, как того требует метка 177, поэтому отбрасывает заголовок MPLS и старается обработать кадр как IP-пакет. Как и в случае исправного пути, адрес 127.0.0.1 требует передачи пакета локальному стеку TCP/IP. Именно этого эффекта и добивались разработчики протокола LSP Ping, выбирая в качестве адреса назначения этот специальный адрес. Узел LSR7 обрабатывает сообщение *Echo Request* и отправляет сообщение *Echo Reply* узлу LSR1 с информацией об обнаруженной ошибке.

Трассировка путей LSP

При неисправном состоянии какого-то отрезка пути LSP сообщение об ошибке не всегда может быть отправлено промежуточным устройством LSP. Возможна и такая ситуация, когда ответ на запрос *Echo Request* просто не приходит — сеть «молчит», например, потому что отказал промежуточный узел. Для того чтобы локализовать отказавший элемент сети (узел или соединение), протокол LSP Ping может работать в режиме трассировки пути LSP. Этот режим аналогичен режиму работы утилиты traceroute стека TCP/IP, и в нем используется тот же механизм, заключающийся в посылке серии сообщений *Echo Request* с монотонно возрастающим от 1 значением поля TTL. Разница состоит в том, что это поле указывается не в IP-пакете, как при использовании IP-утилиты traceroute, а в заголовке MPLS (который также имеет поле TTL).

Дальнейшее поведение протокола LSP Ping в режиме трассировки очевидно — MPLS-кадр с нулевым значением TTL передается «наверх» протоколу LSP Ping того промежуточного узла, который после вычитания единицы из значения этого поля получил нулевой результат. Протокол реагирует на такую ситуацию отправкой сообщения *Echo Reply* начальному узлу тестируемого пути.

Протокол двунаправленного обнаружения ошибок продвижения

Протокол **двунаправленного обнаружения ошибок продвижения** (Biderectional Forwarding Detection, BFD) разработан как «облегченная» альтернатива протоколу LSP Ping для постоянного мониторинга состояния пути LSP. Подобный постоянный мониторинг требуется, например, в тех случаях, когда основной путь защищен резервным путем. То есть необходим некий механизм, который, с одной стороны, мог бы быстро выявить отказ пути, а с другой — не перегружает сеть тестовыми сообщениями и трудоемкими проверками. Протокол LSP Ping удовлетворяет первому условию, то есть может использоваться для постоянного тестирования состояния пути путем периодической отправки сообщений *Echo Requst*. Однако обработка этих сообщений конечным узлом пути довольно трудоемка, так как требует сравнения значения FEC в каждом пришедшем запросе со значением из базы данных.

Протокол BFD гораздо проще, чем LSP Ping. Он не способен локализовать отказавший элемент сети, а только показывает, работоспособен некоторый путь LSP или нет.

Название протокола говорит о том, что он проверяет состояние соединения между двумя узлами в обоих направлениях. Так как пути MPLS однонаправленные, то для работы протокола BFD необходима пара путей LSP, соединяющих два узла в обоих направлениях.

Каждый из двух конечных узлов, на которых для мониторинга определенного пути LSP развернут протокол BFD, периодически посылает по этому пути сообщения *Hello*. Получение сообщений *Hello* от соседа означает работоспособность пути в одном определенном направлении. Неполучение сообщения *Hello* в течение определенного времени означает отказ пути в этом направлении, что и фиксирует протокол BFD. Информацию об отказе пути могут немедленно использовать другие протоколы стека MPLS, например рассматриваемые далее протоколы защиты пути.

Протокол BFD посылает сообщения *Hello* в UDP-сообщениях, которые в свою очередь упаковываются в IP-пакеты и снабжаются заголовками MPLS. Протокол BFD может использоваться не только для мониторинга путей MPLS, он разработан как универсальный протокол тестирования двунаправленных соединений. Обычно для инициализации сеанса BFD служит протокол LSP Ping, который переносит по пути идентификаторы сеанса BFD.

Отказоустойчивость путей в MPLS

Общая характеристика

MPLS поддерживает несколько механизмов обеспечения отказоустойчивости, или, в терминах SDH, механизмов *автоматического защитного переключения* маршрута, в случае отказа какого-либо элемента сети: интерфейса LSR, линии связи или LSR в целом.

В том случае, когда путь устанавливается с помощью протокола LDP, существует единственная возможность защиты пути — его восстановление с помощью распределенного механизма нахождения нового пути средствами протоколов маршрутизации. Это абсолютно тот же механизм, который используется в IP-сетях при отказе линии или маршрутизатора. Время восстановления пути зависит от применяемого протокола маршрутизации и сложности топологии сети, обычно это десятки секунд или несколько минут.

В том случае, когда путь является TE-туннелем, в технологии MPLS разработано несколько механизмов его восстановления. Эти механизмы иллюстрирует рис. 20.15, на котором показан основной путь LSP 1, соединяющий устройства LSR1 и LSR8. Будем считать, что путь LSP1 является TE-туннелем.

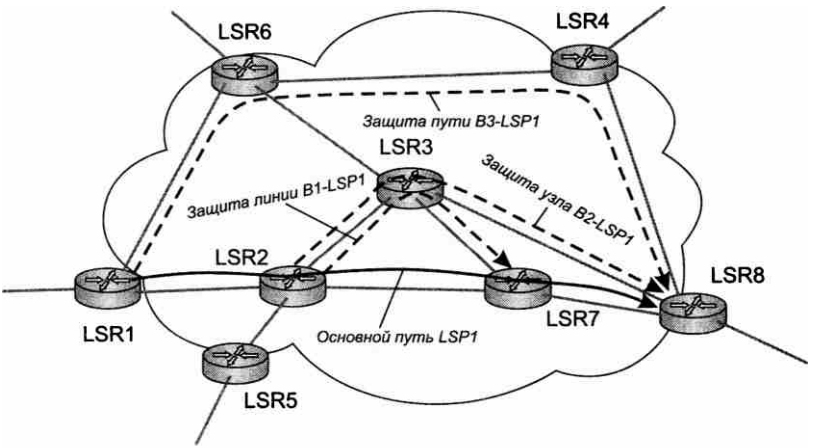


Рис. 20.15. Защитные механизмы MPLS

- *Восстановление пути его начальным узлом* представляет собой традиционное (с помощью протокола маршрутизации) повторное нахождение нового пути, обходящего отказавший элемент сети. Отличие от восстановления пути LDP заключается только в том, что прокладкой нового пути занимается лишь один узел сети, а именно начальный узел пути. В нашем примере это узел LSR1.
- *Защита линии* организуется между двумя устройствами LSR, непосредственно соединенными линией связи. Обходной маршрут находится заранее, до отказа линии, и заранее прокладывается между этими устройствами таким образом, чтобы обойти линию связи в случае ее отказа. В нашем примере такой вариант защиты установлен для линии, соединяющей узлы LSR2 и LSR7. Обходной путь B1-LSP1 проложен через узел LSR3. Защита линии является временной мерой, так как параллельно с началом использования обходного пути начальный узел основного пути начинает процедуру его восстановления с помощью протокола маршрутизации. После восстановления основного пути использование обходного пути прекращается. Временная защита линии не гарантирует TE-туннелю требуемой пропускной способности. Механизм защиты линии работает очень быстро, обычно время переключения не превосходит 50 мс, то есть сравнимо со временем переключения сетей SDH, которые всегда выступают в этой области в качестве эталона. Поэтому механизм защиты линии называют быстрой перемаршрутизацией (fats re-route).
- *Защита узла* очень похожа на защиту линии, отличаясь тем, что обходной путь прокладывается так, чтобы обойти отказавшее устройство LSR (в нашем примере это устройство LSR7). Все остальные характеристики аналогичны характеристикам защиты линии; защита узла тоже относится к механизмам быстрой перемаршрутизации и тоже является временной мерой.

- *Защита пути* организуется так, что в дополнение к основному пути в сети прокладывается путь, связывающий те же конечные устройства, но проходящий по возможности через устройства LSR и линии связи, не встречающиеся в основном пути (на рисунке это резервный путь B3-LSP1). Данный механизм самый универсальный, но он работает медленнее, чем механизмы защиты линии и узла.

Для быстрого обнаружения отказа основного пути или его части могут использоваться различные механизмы и протоколы: сообщения *Hello* протокола RSVP, протокол LSP Ping или BFD.

Использование иерархии меток для быстрой защиты

Рассмотрим работу быстрых механизмов защиты на примере защиты линии, представленной на рис. 20.16. Пусть для защиты линии LSR2-LSR7 в сети проложен обходной путь B-LSP1. На основном пути LSP1 для продвижения кадров используется последовательность меток 15, 17 и 21. На первом участке обходного пути B-LSP1 используется метка 7, на втором — метка 8.

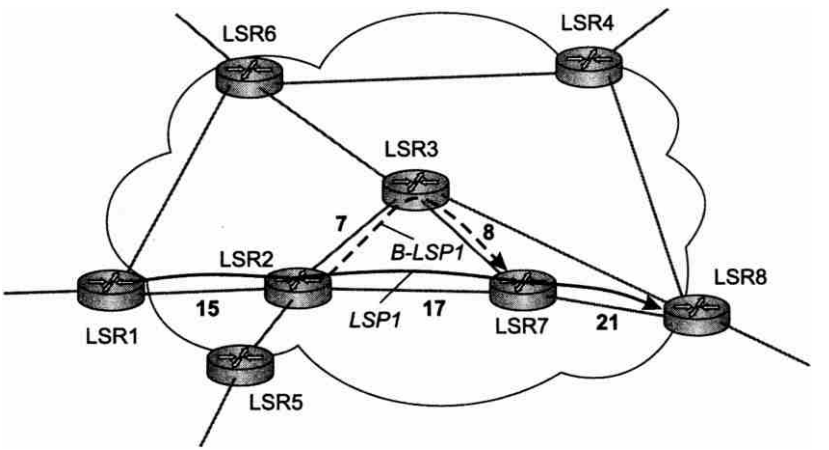


Рис. 20.16. Распределение меток для основного пути и обходного пути защиты линии

При отказе линии LSR2-LSR7 устройство LSR2 начинает направлять в обходной путь B-LSP1 кадры, поступающие по пути LSP1 (рис. 20.17). Однако если при этом поменять метку 15 на метку 7, как того требует обычная логика коммутации меток, то кадр придет в устройство LSR7 с меткой 8 (ее установит устройство LSR3), которая не соответствует значению метки 17, используемой в устройстве LSR7 для передачи кадров по пути LSP1.

Для того чтобы устройство LSR7 работало при переходе на обходной путь точно так же, как и при нормальной работе основного пути, в технике быстрой защиты применяется иерархия меток. Для этого устройство LSR2, которое реализует механизм защиты линии, заменяет метку 15 в пришедшем пакете меткой 17, как если бы линия LSR2-LSR7 оставалась работоспособной. Затем устройство LSR2 проталкивает метку первого уровня в стек, а на вершину стека помещает метку 7, которая нужна для продвижения кадра по обходному пути.

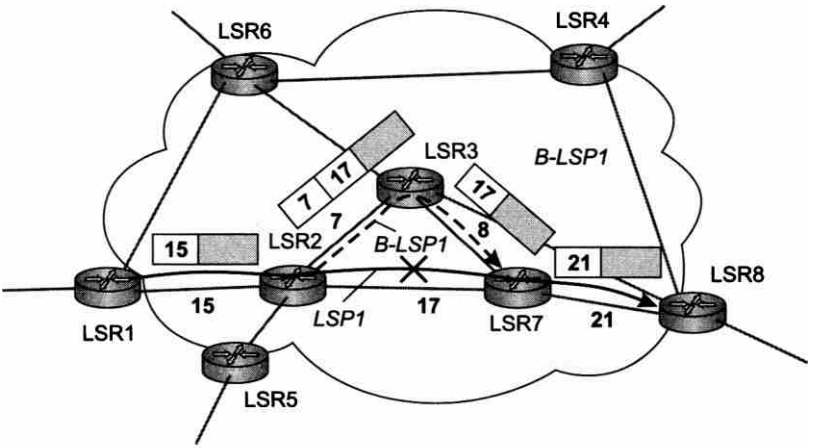


Рис. 20.17. Передача кадров по обходному пути

Устройство LSR3 является предпоследним устройством обходного пути. Поэтому оно удаляет верхнюю метку 7 и выталкивает на вершину стека метку 17. В результате кадр поступает в коммутатор LSR7 с меткой 17, что и требуется для продвижения его далее по пути LSP1.

Аналогичным образом работает механизм быстрой защиты узла, в нем также используется иерархия меток.

Выводы

Технология MPLS считается сегодня многими специалистами одной из самых перспективных транспортных технологий. Главный принцип MPLS: протоколы маршрутизации используются для определения топологии сети, а для продвижения данных внутри границ сети одного поставщика услуг применяется техника виртуальных каналов.

Объединение техники виртуальных каналов с функциональностью стека TCP/IP происходит за счет того, что одно и то же сетевое устройство, называемое коммутирующим по меткам маршрутизатором (LSR), выполняет функции как IP-маршрутизатора, так и коммутатора виртуальных каналов.

Кадры MPLS имеют заголовки двух типов:

- внешний заголовок одной из технологий канального уровня, например Ethernet или PPP;
- заголовок-прокладка с полем метки и некоторыми другими полями, относящимися собственно к технологии MPLS.

MPLS поддерживает иерархию путей за счет применения стека меток. При этом число уровней иерархии не ограничено.

Протокол LDP позволяет автоматически назначать метки для вновь прокладываемого пути LSP. Маршрут для этого пути выбирается с помощью стандартных протоколов маршрутизации.

Для тестирования состояния пути LSP в технологии MPLS разработан протокол LSP Ping, работа которого во многом похожа на работу утилиты ping стека TCP/IP. Мониторинг состояния пути LSP можно выполнять с помощью протокола BFD.

Существует несколько механизмов обеспечения отказоустойчивости в сетях MPLS:

- восстановление пути его начальным узлом;
- защита линии;
- защита узла;
- защита пути.

Технология MPLS поддерживает инжиниринг трафика. Для этого применяются специальные версии протоколов маршрутизации, такие как OSPF TE и IS-IS TE, которые учитывают свободную пропускную способность каждой линии связи сети.

Автоматическое установление найденного в соответствии с задачами инжиниринга трафика пути осуществляется специальной версией протокола RSVP, которая имеет название RSVP TE.

Контрольные вопросы

1. Технология MPLS является гибридом технологий:
   а) IP и IPX;
   б) IP и OSPF;
   в) IP и технологии виртуальных каналов.
2. Какие функциональные модули IP-маршрутизатора используются в LSR? Варианты ответов:
   а) блок продвижения;
   б) блок протоколов маршрутизации;
   в) блок протоколов канального уровня.
3. Класс эквивалентности продвижения — это:
   а) набор путей LSP с равными метриками;
   б) набор путей к одному и тому же выходному устройству LER;
   в) группа IP-пакетов, имеющих одни и те же требования к условиям транспортировки.
4. Протокол LDP позволяет автоматически проложить пути LSP, причем маршруты для них:
   а) определяются стандартной таблицей маршрутизации;
   б) определяются с помощью техники инжиниринга трафика;
   в) учитывают свободную пропускную способность линий связи.
5. Какие узлы пути задаются при описании свободного TE-пути? Варианты ответов:
   а) только конечный;
   б) начальный и конечный;
   в) начальный, конечный и часть промежуточных узлов.

ГЛАВА 21 Ethernet операторского класса

Движущие силы экспансии Ethernet

Как мы знаем, классическая технология Ethernet разрабатывалась исключительно как технология локальных сетей, и до недавнего времени сети этого типа были единственной областью ее применения. Однако бесспорный успех Ethernet в локальных сетях, где она вытеснила все остальные технологии, привел к напрашивающейся идее о ее использовании в глобальных сетях (которые по большей части являются операторскими).

Потенциальных преимуществ от экспансии Ethernet за пределы локальных сетей несколько.

Для пользователей технология Ethernet важна *в качестве услуги глобальных сетей*. Эта услуга может у разных провайдеров называться по-разному — Carrier Ethernet, Ethernet VPN, VPLS, ELINE или ELAN, — суть от этого не меняется: пользователи получают возможность соединять свои территориально рассредоточенные сети, подключая их к интерфейсу Ethernet, предоставляемому провайдером. При этом их сети объединяются так же, как они объединяются в пределах офиса, то есть на уровне Ethernet и без привлечения протокола IP. Это означает, что сеть провайдера учитывает только MAC-адреса, идентификаторы VLAN и физический интерфейс пользователя для того, чтобы надлежащим образом обеспечить объединение сетей пользователя.

При этом пользователи имеют дело с хорошо изученной технологией на интерфейсах доступа к сети провайдера, то есть интерфейсах UNI. Кроме того, при соединении сетей на канальном уровне пользователи свободны в IP-адресации своих сетей, так как при передаче трафика между сетями пользователей услуги Ethernet операторского класса провайдер не применяет IP-адреса. Таким образом, можно, например, назначить адреса одной и той же IP-подсети для всех сетей пользователей или же задействовать частные IP-адреса. Это общее свойство услуг *VPN канального уровня*, но сегодня такая услуга практически всегда выглядит как услуга с интерфейсом Ethernet.

Для провайдеров технология Ethernet операторского класса важна не только как популярная услуга, но и как *внутренняя транспортная технология канального уровня*, в этом случае ее также называют **Carrier Ethernet Transport (CET)** и могут использовать для реализации глобальных услуг Ethernet или же создания надежных, быстрых и контролируемых соединений между маршрутизаторами.

Привлекательность Ethernet как внутренней транспортной технологии для операторов связи объясняется относительно низкой стоимостью оборудования Ethernet. Порты Ethernet всегда обладали самой низкой стоимостью по сравнению с портами любой другой технологии (естественно, с учетом скорости передачи данных портом). Низкая стоимость изначально была результатом простоты технологии Ethernet, которая предлагает только

минимальный набор функций по передаче кадров в режиме доставки по возможности (с максимальными усилиями), не поддерживая ни контроль над маршрутами трафика, ни мониторинг работоспособности соединения между узлами. Низкая стоимость оборудования Ethernet при удовлетворительной функциональности привела к доминированию Ethernet на рынке оборудования локальных сетей, ну а далее начал работать механизм положительной обратной связи: хорошие продажи — массовое производство — еще большее удешевление и т. д.

Стремление к унификации также относится к силам, ведущим к экспансии Ethernet в глобальные сети. Сетевой уровень уже давно демонстрирует однородность благодаря доминированию протокола IP, и перспектива получить однородный канальный уровень в виде Ethernet выглядит очень заманчивой.

Однако все это относится к области желаний, а как обстоит дело с возможностями? Готова ли технология Ethernet к новой миссии? Ответ очевиден: в своем классическом виде технологии локальной сети не готова. Для того чтобы успешно работать в сетях операторов связи, технология и воплощающее ее оборудование должны обладать определенным набором характеристик, среди которых в первую очередь нужно отметить надежность, отказоустойчивость, масштабируемость и управляемость. Эталоном такой технологии может служить технология SDH, которая долгие годы использовалась (и все еще используется) как становой хребет сетей операторов связи, соединяя своим каналами маршрутизаторы, телефонные станции и любое другое оборудование провайдера. Технология MPLS также может выступать в качестве такого эталона, ее основные свойства, описанные в предыдущей главе, позволяют сделать такой вывод.

Области улучшения Ethernet

Рассмотрим более подробно те новые свойства, которые необходимо добавить к классическому варианту Ethernet, чтобы превратить Ethernet в транспортную технологию операторского класса (то есть CET), способную работать в сети провайдера в качестве основного транспортного механизма.

Разделение адресных пространств пользователей и провайдера

Адресное пространство сети современной коммутируемой сети Ethernet состоит из двух частей: значений MAC-адресов конечных узлов и значений идентификаторов локальных виртуальных сетей (VLAN), на которые логически разделена сеть. Коммутаторы Ethernet при принятии решения о продвижении кадра учитывают оба адресных параметра.

Если сеть провайдера будет составлять с сетями пользователей единое целое на уровне Ethernet, то такая сеть окажется практически неработоспособной, так как все коммутаторы провайдера должны будут в своих таблицах продвижения содержать MAC-адреса всех конечных узлов всех пользователей, а также поддерживать принятое каждым пользователем разбиение сети на локальные виртуальные сети. Помимо очевидной проблемы количества MAC-адресов (для крупного провайдера это значение может доходить до нескольких миллионов), есть еще проблема их уникальности — хотя система назначения

адресов и призвана предотвратить дублирование «аппаратных» MAC-адресов, существуют еще и программируемые адреса, да и ошибки в прошивке аппаратных адресов тоже случаются.

Применение в сети провайдера пользовательских идентификаторов VLAN также приводит к проблемам. Во-первых, пользователям нужно договариваться о согласованном применении идентификаторов VLAN, чтобы они были уникальными для каждого пользователя, так как только тогда сеть провайдера сможет доставлять кадры нужным пользовательским сетям. Представить, как реализовать такую процедуру практически, очень непросто, ведь каждый новый пользователь приходит со своими значениями идентификаторов VLAN, и если заставлять его их переназначать, то можно потерять пользователя. Кроме того, стандарт VLAN изначально не был рассчитан на глобальное применение и поэтому в нем предусмотрено только 4092 значения метки, что крайне мало для крупного провайдера.

Если посмотреть, как решаются эти проблемы в сетях провайдеров, построенных на других принципах, то мы увидим, что при применении провайдером технологии IP MAC-адреса пользователей вообще не проникают в маршрутизаторы провайдера[1], а IP-адреса пользователей представлены в таблицах маршрутизаторов в агрегированном виде, — прием, недоступный для плоских MAC-адресов.

Маршрутизация, инжиниринг трафика и отказоустойчивость

Операторы связи привыкли к ситуации полного контроля над путями следования трафика в своих сетях, что обеспечивает, например, технология SDH. В IP-сетях степень контроля оператора над маршрутами трафика очень низкая, и одной из причин популярности технологии MPLS служит то, что она привнесла в IP-сети возможности инжиниринга трафика. Другой желательной для операторов характеристикой сети является отказоустойчивость маршрутов, то есть возможность быстрого перехода на новый маршрут при отказах узлов или линий связи сети. Технология SDH всегда была в этом плане эталоном, так как обеспечивает переход с основного на заранее проложенный резервный путь за десятки миллисекунд. Подобным свойством обладает также технология MPLS.

В сетях Ethernet маршрутизация трафика и отказоустойчивость обеспечиваются протоколом покрывающего дерева (STP). Этот протокол дает администратору сети очень ограниченный контроль над выбором маршрута (это справедливо и для новых вариантов STP, таких как RSTP и MSTP). Кроме того, покрывающее дерево является общим для всех потоков независимо от их адреса назначения. Ввиду этих особенностей протокол STP/RTP является очень плохим решением в отношении инжиниринга трафика. И хотя STP обеспечивает отказоустойчивость маршрутов, причем новая версия RTP значительно сократила время переключения на новый маршрут (с нескольких десятков секунд до одной-двух), до миллисекундного диапазона SDH ей очень далеко. Все это требует нового подхода к маршрутизации потоков в сетях CET, и IEEE работает над этой проблемой.

---

[1] Если быть педантичным, нужно сделать оговорку: за исключением MAC-адресов пограничных интерфейсов пользовательских маршрутизаторов, которые попадают в ARP-таблицы интерфейсов пограничных маршрутизаторов провайдера в случае, если это интерфейсы Ethernet.

Функции эксплуатации, администрирования и обслуживания

Функции эксплуатации, администрирования и обслуживания (Operation, Administration, Maintenance, OAM) всегда были слабым звеном Ethernet, и это одна из главных причин, по которой операторы связи не хотели применять эту технологию в своих сетях. Новые стандарты, предлагаемые IEEE и ITU-T, призваны исправить эту ситуацию, вводя средства, с помощью которых можно выполнять мониторинг достижимости узлов, локализовывать неисправные сегменты сети и измерять уровень задержек и потерь кадров между узлами сети.

В заключение этого обзора областей улучшений функциональности Ethernet нужно отметить, что первая область связана с решением проблемы использования Ethernet для оказания услуги виртуальных частных сетей, а две остальные — с приданием Ethernet функциональности, необходимой для применения Ethernet в качестве внутренней транспортной технологии оператора связи.

Функции OAM в Ethernet операторского класса

Мы начнем рассмотрение улучшений Ethernet с группы функций OAM.

К настоящему времени разработано несколько стандартов, относящихся к функциям эксплуатации, администрирования и обслуживания, необходимых для превращения Ethernet в Ethernet операторского класса:

- IEEE 802.1ag. **Connectivity Fault Management (CFM)**. Стандарт описывает протокол мониторинга состояния соединений, в какой-то степени это аналог протокола BFD, рассмотренного в главе 20.
- ITU-T Y.1731. Стандарт комитета ITU-T воспроизводит функции стандарта IEEE 802.1ag CFM и расширяет их за счет группы функций мониторинга параметров QoS.
- IEEE 802.3ah. Стандарт тестирования физического соединения Ethernet.
- MEF E-LMI. Интерфейс локального управления Ethernet.

Протокол CFM

Протокол CFM обеспечивает мониторинг логических соединений Ethernet. Этот протокол ориентируется на технику виртуальных локальных сетей (VLAN), под логическим соединением в нем понимается соединение узлов, принадлежащих одной сети VLAN. Протокол CFM рассчитан на тестирование соединений любой топологии: двухточечной, звездообразной, полносвязной.

Протокол CFM может выполнять мониторинг как в сети, принадлежащей одному провайдеру (однодоменный сценарий), так и в тех случаях, когда соединение проходит через сети нескольких провайдеров (многодоменный сценарий).

Мониторинг выполняется между так называемыми **конечными точками обслуживания** (Maintenance End Point, **MEP**), представляющими собой конечные точки соединения,

состояние которого нужно наблюдать. Точки MEP располагаются на интерфейсах коммутаторов сети Ethernet, то есть мониторинг выполняется между двумя интерфейсами коммутаторов сети.

Каждая из точек MEP периодически посылает **сообщения проверки непрерывности соединения** (Continuity Check Message, **CCM**), оформленные как кадры сети VLAN, соединения которой тестируются. Например, если наблюдается соединение VLAN 5, то сообщения CCM оформляются как кадры Ethernet с идентификатором VLAN, равным 5. Соединение между точками MEP тестируется отдельно в каждом направлении.

Мониторинг CFM осуществляется путем *активных измерений*, так как для его реализации генерируются служебные сообщения CCM, а не используются кадры пользовательского трафика (см. главу 5).

Устройства, которые не имеют точек MEP, передают сообщения CCM транзитом.

В том случае, когда некоторая точка MEP не принимает сообщения CCM от другой точки MEP в течение заданного *тайм-аута*, соединение считается *неработоспособным*.

В промежуточных устройствах, через которые проходит соединение, можно сконфигурировать **промежуточные точки обслуживания** (Maintenance Intermediate Point, **MIP**). Эти точки помогают локализовать проблему, собирая статистику о проходящих через них транзитом сообщениях CCM, сами они такие сообщения не генерируют. Помощь MIP состоит в том, что при наличии проблемы (то есть в том случае, когда сообщения CCM не проходят от одной точки MEP до другой) факт прохождения сообщений CCM через некоторую точку MIP говорит о том, что данный сегмент сети работоспособен и причину проблемы нужно искать в другом сегменте.

На рис. 21.1 показан пример мониторинга состояния соединения локальной виртуальной сети VLAN 5. Эта сеть имеет полносвязную топологию, поэтому компьютеры C1, C2 и C3 могут взаимодействовать между собой по принципу «каждый с каждым». Для мониторинга сети VLAN5 созданы три точки, MEP1, MEP2 и MEP3, которые располагаются на интерфейсах коммутаторов S1, S2 и S5 соответственно.

Для того чтобы осуществлять мониторинг соединений полносвязной топологии, которую имеет VLAN 5, сообщения CCM посылаются с групповым MAC-адресом. Для мониторинга двухточечных соединений могут использоваться как индивидуальные, так и групповые MAC-адреса.

В нашем примере точка MEP1 периодически посылает в сеть VLAN5 сообщения CCM с групповым адресом. Если сеть VLAN5 работоспособна, то точки MEP2 и MEP3 регулярно получают сообщения CCM, отправляемые точкой MEP1. Также регулярно передаются и принимаются сообщения CCM, генерируемые точками MEP2 и MEP3. В результате протокол CFM определяет статус сети VLAN5 как полностью работоспособной.

Предположим теперь, что в сети произошел отказ физического соединения между коммутаторами S4 и S5. Вследствие этого точка MEP3 перестает принимать сообщения CCM от точек MEP1 и MEP2, а они, в свою очередь, — сообщения CCM от точки MEP3. В то же время точки MEP1 и MEP2 по-прежнему продолжают обмениваться сообщениями CCM. Результатом мониторинга будет переход соединения VLAN5 в состояние частичной работоспособности, когда только часть узлов оказывается достижимой.

Покажем теперь полезность наличия MIP в сети. Предположим, что в сети, показанной на рис. 21.1, связь между коммутаторами S4 и S5 восстановлена, но по какой-то причине

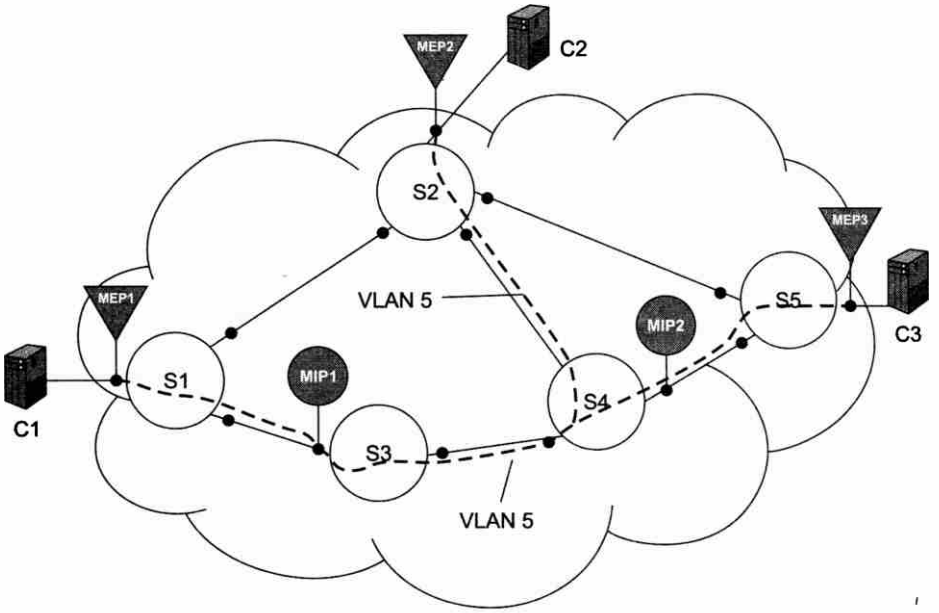


Рис. 21.1. Мониторинг состояния VLAN с помощью протокола CFM

потеряна связь между коммутаторами S3 и S4. Точка MEP1 при этом перестает принимать сообщения от точки MEP3, а точка MEP3 — от точки MEP1 (для упрощения анализа мы сейчас игнорируем точку MEP2 и контролируемую ею часть сети). Ясно, что точки MEP1 и MEP3 фиксируют нарушение связности между собой, но их информация не позволяет судить о том, где конкретно в сети возникла проблема, — отказ мог произойти в любом из трех сегментов сети между коммутаторами S1 и S5. Но так как в сети имеются точки MIP, то администратор может проанализировать их статистику. Статистика MIP1 покажет, что через эту точку по-прежнему проходят сообщения CCM от MEP1, но не проходят сообщения MEP3. Статистика MIP2 покажет обратную картину — наличие сообщений от MEP3, но не от MEP1. Эти данные свидетельствуют о том, что связность потеряна между коммутаторами S3 и S4.

Весьма важной является способность протокола CFM работать в *многодоменной среде*, когда соединение проходит через несколько сетей, принадлежащих разным административным доменам. Каждый из администраторов домена нуждается в мониторинге соединения, но только в пределах своей сети.

Для поддержки мониторинга состояния соединений в многодоменной сети для протокола CFM конфигурируется отдельный *домен мониторинга*, при этом домены мониторинга образуют иерархию доменов различного *уровня мониторинга*. В каждом домене создаются точки обслуживания MEP и MIP, но точки каждого домена работают только с сообщениями CCM своего уровня, а сообщения более высоких уровней просто прозрачно передают.

Эту идею иллюстрирует рис. 21.2. Здесь показана сеть, состоящая из доменов различных типов: домена пользователя, домена провайдера услуги виртуальной частной сети и до-

мена оператора связи, через который работает сеть провайдера услуги. В сети имеется три домена операторов: оператора A, оператора B и оператора C. Домены операторов вложены в домен провайдера услуг, который предоставляет пользователю услуги виртуальной локальной сети «из конца в конец». И наконец, на верхнем уровне находится домен пользователя, в который входит домен провайдера услуги.

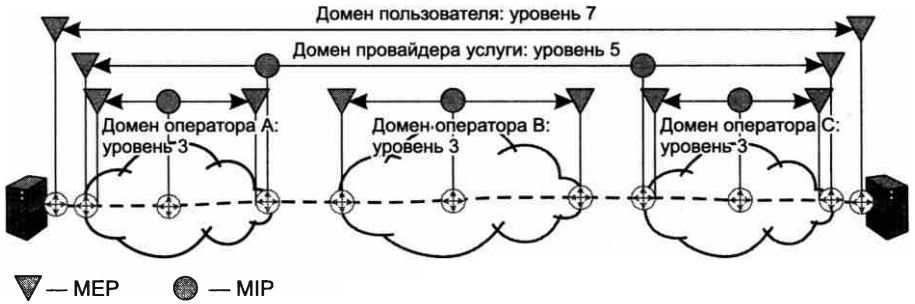


Рис. 21.2. Многодоменное применение протокола CFM

Домену пользователя присвоен уровень 7, домену провайдера — уровень 5, домену оператора связи — уровень 3. Точки MEP на интерфейсах оборудования оператора связи работают с сообщениями CCM уровня 3, а сообщения точек обслуживания сети пользователя уровня 7 и сети провайдера услуги уровня 5 они передают прозрачно. Аналогично, точки провайдера услуги работают на интерфейсах его оборудования на уровне 5, они прозрачно передают сообщения CCM уровня домена пользователя 7. Сообщения уровня 3 до точек MEP уровня 5 не доходят, так как завершаются в точках MEP уровня 3.

В результате каждый оператор связи получает информацию о состоянии соединения в пределах своей сети, провайдер — в пределах своей, а пользователь соединения — «из конца в конец». Такой иерархический способ организации сеансов между точками MEP дает возможность проводить независимый мониторинг одного и того же соединения различными организациями без необходимости координировать конфигурацию точек мониторинга — достаточно согласовать используемые каждой организацией уровни точек MEP. Обязательным условием является вложенность доменов каждого уровня в домен более высокого уровня иерархии.

Протокол мониторинга качества соединений Y.1731

Стандарт Y.1731, разработанный ITU-T, добавляет к стандарту CFM возможность измерять некоторые дополнительные параметры между точками мониторинга сети.

- *Односторонняя задержка кадра*. Для измерения этой задержки точки обслуживания сети MEP генерируют сообщения измерения задержки и ответа на измерение задержки. В этих сообщениях переносятся временны́е отметки, позволяющие измерить задержку.
- *Вариация задержки*. Эта задержка измеряется на основе тех же сообщений, что и односторонняя задержка.

- *Потери кадров.* Для измерения этой величины служат сообщения измерения потерь и ответа на измерение потерь. Счетчики сообщений двух точек обслуживания сравниваются, и на основе этого сравнения рассчитываются потери кадров в каждом из направлений.

Стандарт тестирования физического соединения Ethernet

Стандарт тестирования физического соединения Ethernet IEEE 802.3ah предназначен для обнаружения ошибок соединения между двумя непосредственно физически связанными интерфейсами Ethernet. Он поддерживает такие функции, как удаленное обнаружение неисправностей и удаленный контроль обратной связи.

Последняя функция является наиболее интересной для специалистов, занимающихся эксплуатацией сетей Ethernet, так как она позволяет удаленно (через сеть) выдать запрос некоторому интерфейсу Ethernet на переход в режим обратной связи. В этом режиме все кадры, посылаемые на этот интерфейс соседом по линии связи, возвращаются им обратно. Полученные кадры затем можно проанализировать, чтобы установить качество физической линии.

Необходимо отметить, что процедура тестирования линии в режиме обратной связи нарушает нормальную работу соединения, поэтому тестирование нужно проводить в специальное время, отведенное для обслуживания сети.

Интерфейс локального управления Ethernet

Стандарт E-LMI позволяет пограничному пользовательскому устройству, то есть устройству типа CE, запрашивать информацию о состоянии и параметрах услуги, предоставляемой сетью провайдера по данному интерфейсу. Например, пограничный коммутатор Ethernet, расположенный в сети пользователя, может запросить у пограничного коммутатора провайдера информацию о состоянии услуги (работоспособности соединения), предоставляемой по данному интерфейсу. Кроме того, согласно стандарту E-LMI, по запросу можно получить такую информацию об услуге, как отображение идентификатора VLAN пользователя на данное соединение, или же величину пропускной способности, гарантированной для данного соединения.

Мосты провайдера

Стандарт IEEE 802.1ad на **мосты провайдера** (Provider Bridge, **PB**) был первым стандартом, который решал проблему изоляции адресного пространства сети провайдера от адресного пространства его пользователей. Этот стандарт был принят IEEE в 2005 году и сегодня реализован в коммутаторах Ethernet многих производителей.

Нужно сказать, что проблема изоляции адресных пространств решается в этом стандарте только частично, так как MAC-адреса пользователей по-прежнему присутствуют в коммутаторах сети провайдера, разделяются только пространства идентификаторов VLAN.

Стандарт PB вводит двухуровневую иерархию идентификаторов VLAN (рис. 21.3). На внешнем (верхнем) уровне располагается идентификатор VLAN провайдера, называемый **S-VID** (от Service VLAN ID — идентификатор сервиса VLAN), а на нижнем (внутреннем) уровне — идентификатор VLAN пользователя, называемый **C-VID** (от Customer VLAN ID — идентификатор VLAN потребителя).

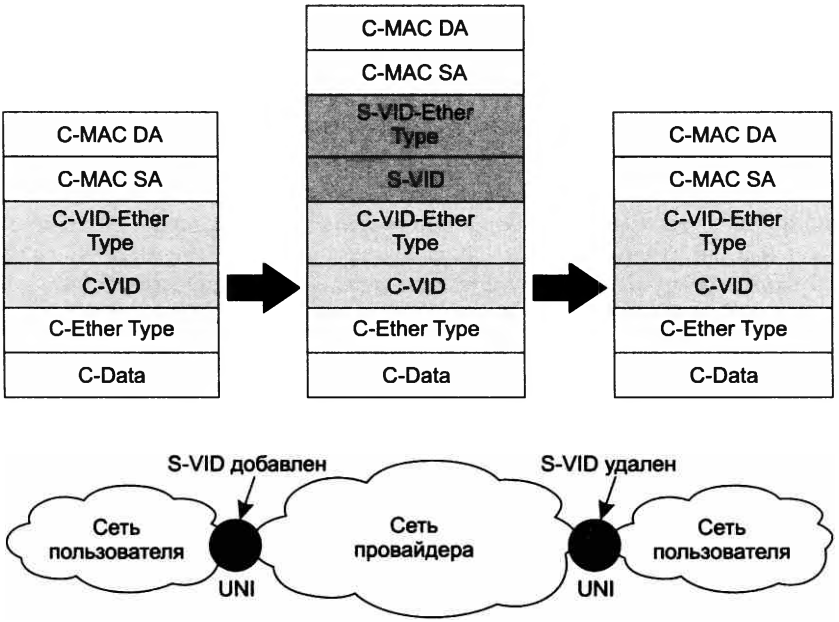


Рис. 21.3. Инкапсуляция идентификаторов VLAN

Идентификатор S-VID помещается в пользовательский кадр пограничным коммутатором провайдера — он проталкивает C-VID в стек и добавляет новый идентификатор S-VID, который потребуется коммутаторам сети провайдера для разделения трафика на виртуальные локальные сети провайдера. Так как S-VID представляет собой новое поле кадра Ethernet, то ему предшествует новое поле типа EtherType, которое на рис. 21.3 обозначено как S-VID-EtherType (в отличие от оригинального поля C-VID-EtherType). Чтобы различать S-VID и C-VID, стандарт 802.1ad вводит новое значение EtherType для типа данных S-VID, равное 0x88a8 (напомним, что для C-VID значение EtherType равно 0x8100). Этот способ инкапсуляции часто неформально называют инкапсуляцией **Q-in-Q** по названию стандарта 802.1Q, описывающего технику VLAN.

После того как пограничный коммутатор сети провайдера выполняет инкапсуляцию Q-in-Q, кадр обрабатывается магистральными коммутаторами провайдера как обычный кадр, то есть в соответствии с внешним идентификатором VLAN в поле S-VID.

Когда кадр прибывает на выходной пограничный коммутатор провайдера, над ним выполняется обратная операция — идентификатор S-VID удаляется. После этого кадр отправляется в сеть пользователя в исходном виде, имея в своем заголовке только идентификатор C-VID.

Внутренние сети VLAN провайдера, соответствующие значениям идентификаторов S-VID, обычно служат для конструирования услуг частных виртуальных сетей. При этом провайдеру нет необходимости согласовывать логическую структуру своей сети с пользователями.

На рис. 21.4 показана сеть провайдера, которая соединяет **сайты** (так обычно называют сети пользователя при оказании услуги VPN) двух пользователей. Сайты C1, C3 и C5 являются сайтами пользователя A, они объединяются в сеть с идентификатором S-VID, равным 156, а сайты C2, C4 и C6 являются сайтами пользователя B, они объединяются в сеть с идентификатором S-VID, равным 505.

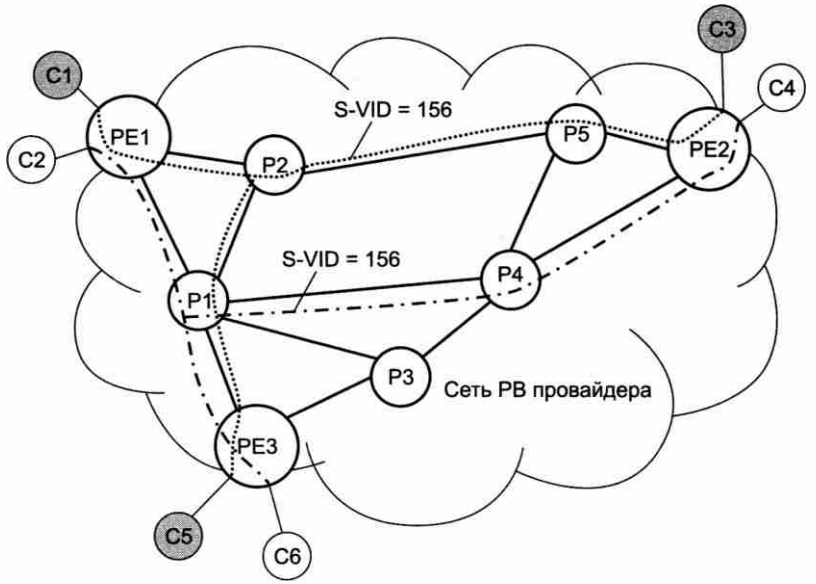


Рис. 21.4. Сеть стандарта PB, предоставляющая услуги соединения сайтов двух пользователей

Конфигурирование услуг сетей 156 и 505 выполнено без учета значений пользовательских идентификаторов VLAN на основании подключения сайта пользователя к некоторому физическому интерфейсу коммутатора провайдера. Так, например, весь пользовательский трафик, поступающий от сайта C1, классифицируется пограничным коммутатором PE1 как принадлежащий виртуальной частной сети с идентификатором S-VID, равным 156.

В то же время стандарт PB позволяет провайдеру предоставлять услуги и с учетом значений пользовательских идентификаторов VLAN. Например, если внутри сайта C1 выполнена логическая структуризация и существуют две пользовательские сети VLAN, трафик которых нельзя смешивать, провайдер может организовать для этого две сети S-VLAN и отображать на них поступающие кадры в зависимости от значений C-VID.

При своей очевидной полезности стандарт PB имеет несколько недостатков.

- Коммутаторы сети провайдера, как пограничные, так и магистральные, должны изучать MAC-адреса узлов сетей пользователей. Это не является масштабируемым решением.

- Максимальное количество услуг, предоставляемых провайдером, ограничено числом 4096 (так как поле S-VID имеет стандартный размер 12 бит).
- Инжиниринг трафика ограничен возможностями протокола покрывающего дерева RSTP/MSTP.
- Для разграничения деревьев STP, создаваемых в сетях провайдера и пользователей, в стандарте 802.1ad пришлось ввести новый групповой адрес для коммутаторов провайдера. Это обстоятельство не позволяет задействовать в качестве магистральных коммутаторов провайдера те коммутаторы, которые не поддерживают стандарт 802.1ad.

Некоторые из этих недостатков были устранены в стандарте на магистральные мосты провайдера IEEE 802.1ah, который был принят летом 2008 года.

Магистральные мосты провайдера

В стандарте IEEE 802.1ah на **магистральные мосты провайдера** (Provider Backbone Bridges, **PBB**) адресные пространства пользователей и провайдера разделяются за счет того, что пограничные коммутаторы провайдера полностью инкапсулируют пользовательские кадры Ethernet в новые кадры Ethernet, которые затем применяются в пределах сети провайдера для доставки пользовательских кадров до выходного пограничного коммутатора.

Формат кадра PBB

При передаче кадров Ethernet через сеть PBB в качестве адресов назначения и источника используются MAC-адреса пограничных коммутаторов (Backbone Edge Bridges, BEB) провайдера. По сути, в сети провайдера работает независимая иерархия Ethernet со своими MAC-адресами и делением сети на виртуальные локальные сети (VLAN) так, как это удобно провайдеру. Из-за двух уровней MAC-адресов в кадрах провайдера стандарт PBB получил также название **MAC-in-MAC**.

Формат кадра при такой инкапсуляции показан на рис. 21.5. Здесь предполагается, что сеть PBB провайдера принимает кадры от сетей PB (возможно, другого провайдера), которые в свою очередь соединены с сетями пользователя. В этом случае в поступающих на пограничные коммутаторы сети PBB кадрах имеется идентификатор S-VID, добавленный входным пограничным коммутатором сети PB. Наличие идентификатора S-VID во входных кадрах не является необходимым условием работы сети PBB, это только возможный вариант; если сеть PBB непосредственно соединяет сети пользователей, то входящие кадры поля S-VID не имеют. Поле S-VID не используется при продвижении кадров в сети PBB, как будет показано далее.

Входной пограничный коммутатор сети PBB добавляет к принимаемому кадру шесть новых полей, из которых четыре поля представляют собой стандартный заголовок нового кадра, в поле данных которого упакован принятый кадр. В этом заголовке MAC-адресами назначения и источника являются адреса выходного и входного пограничных коммутаторов сети, которые на рис. 21.5 обозначены как B-MAC DA и B-MAC SA соответственно (буква «B» в этих обозначениях появилась от слова «backbone» — магистральный). Адреса

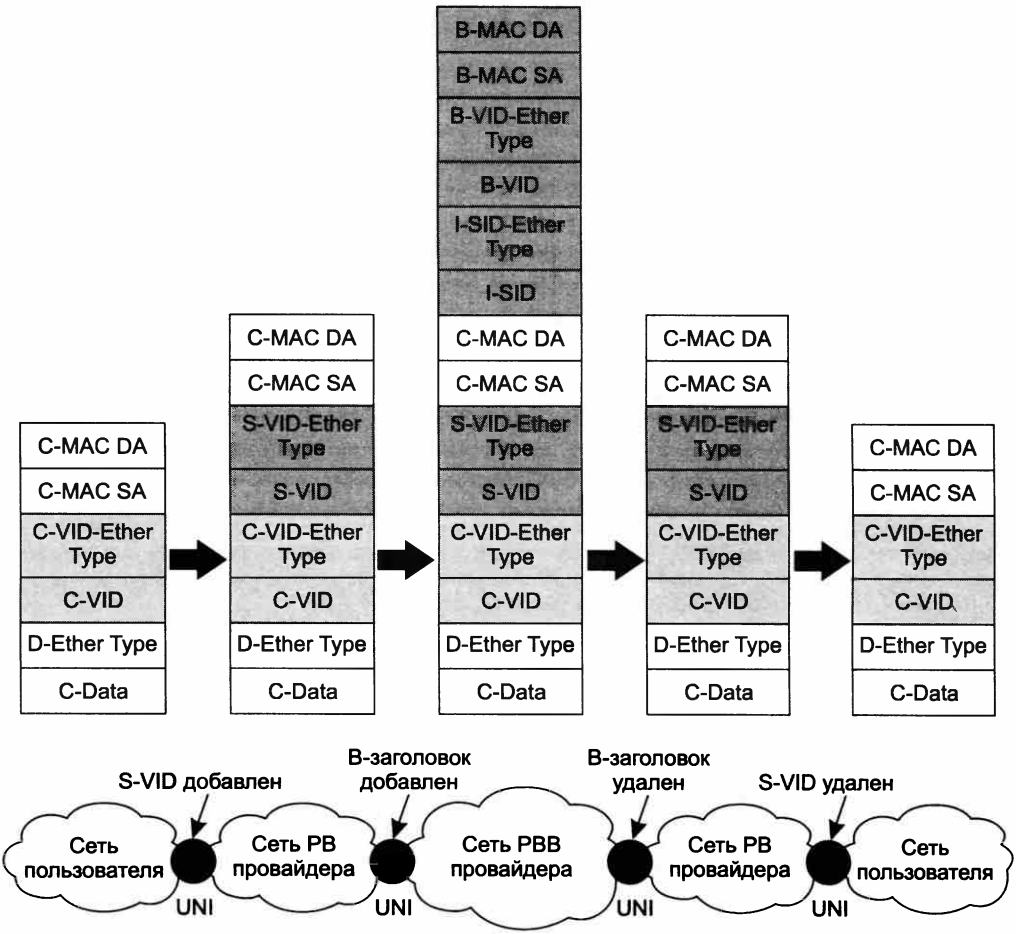


Рис. 21.5. Формат кадров при инкапсуляции MAC-in-MAC согласно стандарту IEEE 802.1ah

B-MAC идентифицируют коммутатор провайдера в целом как узел, являясь аналогом IP-адреса обратной связи маршрутизатора.

Адреса B-MAC используются в пределах сети PBB вместе с идентификатором виртуальной локальной сети B-VID для передачи кадров в соответствии со стандартной логикой локальной сети, разделенной на сегменты VLAN, и при этом совершенно независимо от адресной информации сетей пользователя. В качестве значения EtherType для B-VID стандарт PBB рекомендует применять значение 0x88a8, как и для S-VID в стандарте PB, но допустимы и другие значения, например стандартное для C-VID значение 0x8100 (как и для сетей PB, эта возможность зависит от решения производителя оборудования).

Пользовательские MAC-адреса, а также идентификаторы S-VID и C-VID находятся в поле данных нового кадра и при передаче между магистральными коммутаторами сети PBB никак не используются.

Двухуровневая иерархия соединений

Полная инкапсуляция приходящих кадров не является единственным новшеством стандарта на PBB. Другим усовершенствованием этого стандарта является введение двухуровневой иерархии соединений между пограничными коммутаторами. Это служит обеспечению масштабируемости технологии при обслуживании большого количества пользовательских соединений.

Для этого в кадр PBB введено поле I-SID с предшествующим ему полем I-SID EtherType (с рекомендованным значением 0x88e7). Значение идентификатора I-SID (Information Service Identificator — идентификатор информационного сервиса) должно указывать на *пользовательское соединение* (*виртуальную частную сеть пользователя*) *в сети PBB*. Так как сеть PBB делится на сегменты B-VLAN, то соединения I-SID являются логическими соединениями внутри этих сегментов. Роль сегментов B-VLAN состоит в предоставлении транспортных услуг соединениям I-SID, они являются своего рода туннелями. В каждой сети B-VLAN может насчитываться до 16 миллионов соединений I-SID (это значение определяется форматом поля I-SID, состоящего из 24 разрядов).

Двухуровневый механизм B-VID/I-SID рассчитан на то, что в сети провайдера будет небольшое количество сегментов B-VLAN, которые направляют потоки пользовательских данных, идущих по логическим соединениям I-SID, по нужным маршрутам.

Назначение идентификатора I-SID в сети PBB аналогично назначению идентификатора S-VID в сети PB — оба определяют виртуальную сеть пользователя в сети провайдера. Этот факт объясняет также необязательность наличия поля S-VID в кадрах пользователя, поступающих на входные интерфейсы сети PBB, — это поле является только *одним из признаков*, учитываемых при отображении кадров пользователя на некоторую виртуальную сеть пользователя, существующую в сети провайдера. Если поле S-VID в кадрах пользователя отсутствует, то для отображения используются другие признаки: MAC-адреса, значение поля C-VID или номер интерфейса, с которого поступают кадры пользователя.

На рис. 21.6 показана сеть провайдера, оказывающая услуги Ethernet своим клиентам на основе стандарта на PBB. Она состоит из пограничных коммутаторов (Backbone Edge Bridge, BEB) и магистральных коммутаторов (Backbone Core Bridge, BCB).

Провайдер в этом примере предоставляет услуги трех виртуальных сетей:

- LAN1 — передает голосовой трафик между сетями C1 и C3 (двухточечная топология);
- LAN2 — передает голосовой трафик между сетями C2 и C4 (двухточечная топология);
- LAN3 — передает эластичный трафик данных между сетями C2, C4 и C6 (полносвязная топология).

Пользовательские сети непосредственно подключены к сети PBB, промежуточных сетей PB в этом примере нет.

На верхнем уровне структуризации сети провайдера в ней сконфигурированы две *магистральные виртуальные локальные сети* (B-VLAN) с идентификаторами 1007 и 1033 (обозначены как B-VID 1007 и B-VID 1033 соответственно). В нашем примере различные сети B-VLAN призваны поддерживать трафик разного типа: B-VLAN 1007 поддерживает более требовательный голосовой трафик, а B-VLAN 1033 — менее требовательный эластичный

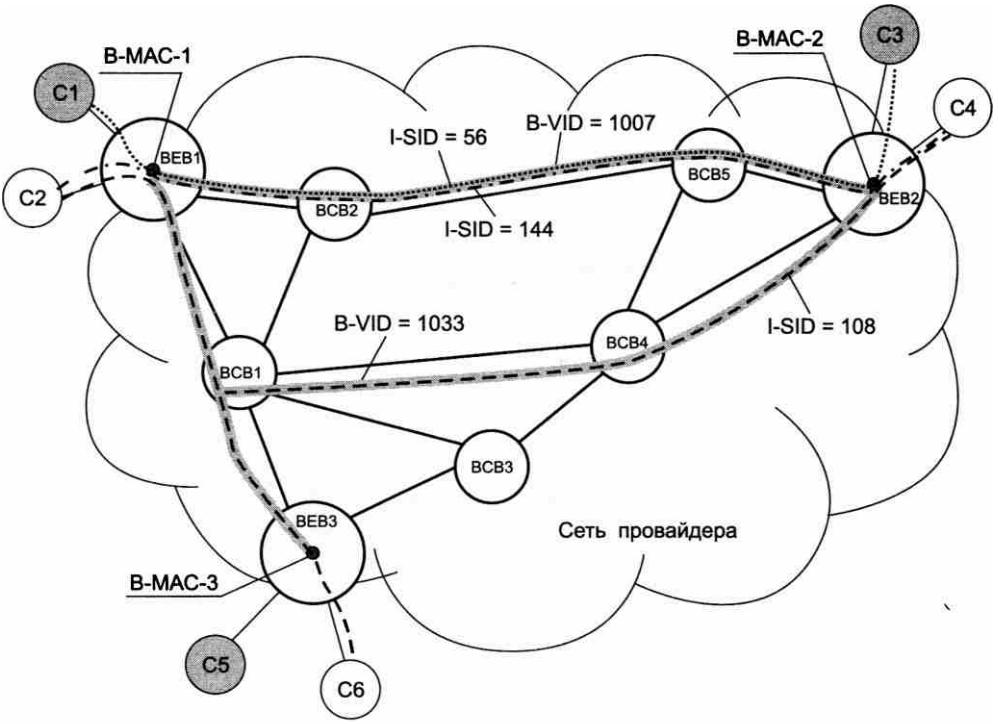


Рис. 21.6. Организация услуг в сети PBB

трафик данных. В соответствии с этим назначением созданы и два покрывающих дерева для каждой из виртуальных сетей B-VLAN. Естественно, что назначение сетей B-VLAN может быть иным — оно полностью определяется оператором сети PBB в соответствии с его потребностями.

На уровне пользовательских услуг в сети организовано три пользовательских соединения, помеченные как I-SID 56, 144 и 108. Эти соединения предназначены для реализации услуг LAN1, LAN2 и LAN3 соответственно.

Соединения I-SID 56 и 144 отображаются пограничными коммутаторами BEB1 и BEB2 на сеть B-VLAN 1007, так как эти соединения переносят пользовательский голосовой трафик, а данная сеть B-VLAN создана для этого типа трафика. В то же время соединение I-SID 108 отображается пограничными коммутаторами BEB1, BEB2 и BEB3 на сеть B-VLAN 1033, так как сервис 108 переносит эластичный пользовательский трафик данных. Задает эти отображения администратор при конфигурировании пограничных коммутаторов.

Завершает процесс конфигурирования услуг LAN1, LAN2 и LAN3 отображение пользовательского трафика на соответствующие соединения I-SID. Это отображение также задает администратор сети при конфигурировании пограничных коммутаторов BEB.

При отображении пользовательского трафика администратор может учитывать только интерфейс, по которому трафик поступает в сеть провайдера. В нашем примере таким способом задано отображение для сервиса с I-SID 56, который монопольно использует интерфейсы коммутаторов BEB1 и BEB2, не разделяя их с другими сервисами.

В том случае, когда на один и тот же интерфейс поступает трафик более чем одного сервиса, при отображении нужно также учитывать значение C-VID (в нашем примере поле S-VID в пользовательских кадрах отсутствует, так как пользовательские сети соединены с сетью PBB непосредственно, без промежуточной сети типа PB). Этот случай имеет место для сервисов с I-SID 144 и 108, так как они разделяют один и тот же интерфейс коммутаторов BEB1 и BEB2. Поэтому такие отображения нужно конфигурировать с учетом значений C-VID: C-VID 305 отображается на I-SID 144, а C-VID 500 — на I-SID 108.

Пользовательские MAC-адреса

Теперь рассмотрим вопрос применения пользовательских MAC-адресов в сети PBB.

Магистральным коммутаторам сети PBB *знание пользовательских адресов не требуется*, так как они передают кадры только на основании комбинации B-MAC/B-VID. А вот поведение пограничных коммутаторов в отношении пользовательских MAC-адресов зависит от топологии сервиса, предоставляемого сетью PBB своим пользователям.

При отображении кадров сервиса с двухточечной топологией на определенное соединение I-SID пограничные коммутаторы не применяют пользовательские MAC-адреса, так как все кадры, независимо от их адресов назначения, передаются одному и тому же выходному пограничному коммутатору. Например, для сервисов с I-SID 56 и 144 коммутатор BEB1 всегда задействует MAC-адрес коммутатора BEB2 в качестве B-MAC DA при формировании несущего (нового) кадра, который переносит инкапсулированный пользовательский кадр через сеть PBB.

Однако при отображении кадров сервиса со звездообразной или полносвязной топологией у входного коммутатора всегда существует несколько выходных пограничных коммутаторов, поддерживающих этот сервис. Например, у входного коммутатора BEB1 при обслуживании кадров сервиса I-SID 108 есть альтернатива — отправить пришедший кадр коммутатору BEB2 или BEB3.

Для принятия решения в таких случаях используется информация, находящаяся в пользовательских MAC-адресах. Пограничные коммутаторы BEB, поддерживающие сервисы со звездообразной и полносвязной топологиями, изучают пользовательские MAC-адреса и посылают кадр выходному коммутатору, связанному с той сетью пользователя, в которой находится MAC-адрес назначения C-MAC DA.

Так, в нашем примере коммутатор BEB1 изучает адреса C-MAC SA кадров, поступающих через сервис I-SID 108, чтобы знать, подключены ли узлы с этими адресами к BEB2 или BEB3. В результате BEB1 создает таблицу продвижения (табл. 21.1).

Таблица 21.1. Таблица продвижения для сервиса I-SID 108

C-MAC	I-SID	B-MAC	B-VID
C-MAC-1	108	B-MAC-2	1033
C-MAC-2	108	B-MAC-2	1033
C-MAC-3	108	B-MAC-3	1033
C-MAC-4	108	B-MAC-3	1033
...	108	...	1033

На основании этой таблицы коммутатор BEB1 по адресу назначения C-MAC выбирает соответствующий адрес выходного пограничного коммутатора и помещает его в формируемый кадр: например, для кадра с адресом назначения C-MAC-2 это будет B-MAC-2. В том же случае, когда пользовательский адрес назначения еще не изучен, коммутатор BEB1 помещает в поле B-MAC широковещательный адрес. Таким же образом обрабатываются кадры с широковещательным пользовательским адресом.

Маршрутизация и отказоустойчивость в сетях PBB

Для нормального функционирования сети PBB ее активная топология должна быть свободна от петель, при этом сеть должна обеспечивать отказоустойчивость, то есть топология сети должна автоматически изменяться в случае отказов линий связи или коммутаторов сети.

В сетях Ethernet для этой цели применяется протокол покрывающего дерева STP, он же может быть применен и в сетях PBB в его версии MSTP, строящей отдельное дерево для каждой магистральной сети VLAN (определяемой значением B-VID). Как вы знаете из материала главы 13, у протокола STP есть несколько принципиальных недостатков, таких как неоптимальность маршрутов и слишком длительное время установления новой активной топологии.

Для преодоления недостатков протокола STP рабочей группой IEEE 802.1aq был создан протокол маршрутизации с **коммутацией по кратчайшему пути** (Shortest Path Bridging, **SPB**). Он основан на протоколе IS-IS, который представляет собой протокол маршрутизации, учитывающий состояние связей (см. главу 17).

Выбор протокола IS-IS для применения в сетях Ethernet объясняется тем, что он создавался как гибкий протокол маршрутизации, способный работать в различных стеках протоколов. Протокол IS-IS может передавать свои сообщения непосредственно в кадрах канального уровня, не используя пакеты IP и сообщения TCP или UDP. Кроме того, для идентификации связей сети он может использовать адресную информацию разного типа. В том случае, когда IS-IS работает в сети IP, он применяет для идентификации связи IP-адреса ее конечных точек. При работе в сети Ethernet IS-IS (точнее, сети SPB, работающей на основе IS-IS) использует для этой цели MAC-адреса.

Группа 802.1aq разработала два варианта протокола SPB: SPBV (SPB VLAN) и SPBM (SPB MAC). Вариант SPBV предназначен для пользовательских сетей, то есть сетей с общим пространством MAC-адресов и одним уровнем виртуальных локальных сетей. Вариант SPBM предназначен для сетей PBB с инкапсуляцией MAC-in-MAC, которая и фигурирует в названии.

При работе протокола SPBM каждый пограничный коммутатор BEB строит дерево оптимальных маршрутов к остальным пограничным коммутаторам отдельно для каждой магистральной сети VLAN, то есть отдельно для каждого значения B-VID. Например, на рис. 21.6 пограничный коммутатор BEB1 строит для сервиса 1033 дерево маршрутов к пограничным коммутаторам BEB2 и BEB3, а для сервиса 1007 — дерево маршрутов только к BEB2, так как только этот коммутатор входит в магистральную виртуальную локальную сеть 1007 кроме BEB1.

Нахождение оптимальных маршрутов выполняется стандартным для протоколов маршрутизации, учитывающих состояние связей, способом: каждый коммутатор (как типа

BEB, так и типа BCB) рассылает объявления о состоянии связей {B-MAC1, B-MAC2}, где B-MAC-адреса относятся к коммутаторам, являющимся конечными точками данной связи. Пограничные коммутаторы BEB распространяют также информацию о номерах магистральных виртуальных сетей B-VID, для которых эти коммутаторы являются конечными. Например, коммутатор BEB1 распространяет информацию о двух связях: {B-MAC1, B-MAC-BCB1} и {B-MAC1, B-MAC-BCB2} (вторые адреса в парах принадлежат магистральным коммутаторам BCB1 и BCB2). Кроме того, он объявляет о том, что является пограничным коммутатором для B-VID 1033 и B-VID 1007.

После получения информации о топологии сети каждый коммутатор сети строит дерево оптимальных маршрутов от себя до каждого конечного коммутатора BEB в каждой магистральной виртуальной сети B-VID. В нашем примере BEB1 строит два дерева: для B-VID 1033 к коммутаторам BEB2 и BEB3, а также для B-VID 1007 к коммутатору BEB2 (вырожденное дерево с одной ветвью). Далее эти деревья служат для построения таблицы продвижения, то есть нахождения следующего хопа передачи кадра. Таблица строится точно так же, как и таблица маршрутизации в протоколах OSPF и IS-IS, то есть выбирается следующий коммутатор вдоль пути к коммутатору назначения.

Применение известного протокола маршрутизации IS-IS как основы протокола SPBM наделяет сети PBB положительными свойствами, характерными для сетей IP, в которых работают протоколы маршрутизации, учитывающие состояние связей, — рациональными маршрутами с отсутствием петель и быстрой перестройкой активной топологии при отказах элементов сети.

Магистральные мосты провайдера с поддержкой инжиниринга трафика

Технология **PBB TE** (Provider Backbone Bridge Traffic Engineering — магистральные мосты провайдера с поддержкой инжиниринга трафика) базируется на технологии PBB, но добавляет к ней возможность инжиниринга трафика. В PBB-TE применяется та же самая схема инкапсуляции кадров, создания магистральных сетей VLAN (B-VID) и пользовательских соединений I-SID. В отличие от PBB, технология PBB-TE работает только с двухточечной топологией соединений.

Главными целями разработчиков технологии PBB TE были:

- **поддержка функций инжиниринга трафика;**
- **обеспечение «быстрой» отказоустойчивости со скоростью, сравнимой со скоростью защиты соединений в технологии SDH.**

Поставленные цели достигаются в технологии PBB TE за счет перечисленных далее изменений технологии PBB и классической технологии локального моста:

- Запрет на работу протокола STP.
- Отключение механизма автоматического изучения магистральных MAC-адресов.
- Использование пары «B-VID/B-MAC-DA» в качестве метки туннеля между двумя пограничными коммутаторами. В принципе, любой коммутатор, который поддерживает технику VLAN (стандарт IEEE 802.1Q), продвигает кадры на выходной порт,

анализируя два указанных в кадре значения: MAC-адрес назначения и идентификатор VLAN. Поэтому данное свойство просто предполагает, что коммутатор ведет себя в соответствии с алгоритмом продвижения, описанным в стандарте 802.1Q, но только для магистральных адресов и магистральных виртуальных локальных сетей.

- Предварительная прокладка первичного (основного) и резервного туннелей для тех случаев, когда нужно обеспечить отказоустойчивость туннеля.

Первые три перечисленных свойства технологии PBB TE позволяют администратору или системе управления сетью формировать пути прохождения через сеть произвольным образом, независимо от того, обеспечивают ли они кратчайшее расстояние (в некоторой метрике) до некоторого коммутатора, названного корневым, или нет, то есть обеспечивают ли поддержку функций инжиниринга трафика. Пара «B-VID/B-MAC-DA» является аналогом метки пути LSP в технологии MPLS, однако в отличие от метки MPLS *значение этой метки остается неизменным* в процессе перемещения кадра по сети провайдера, то есть коммутации метки не происходит.

Посмотрим, как работает технология PBB TE, на примере сети, изображенной на рис. 21.7.

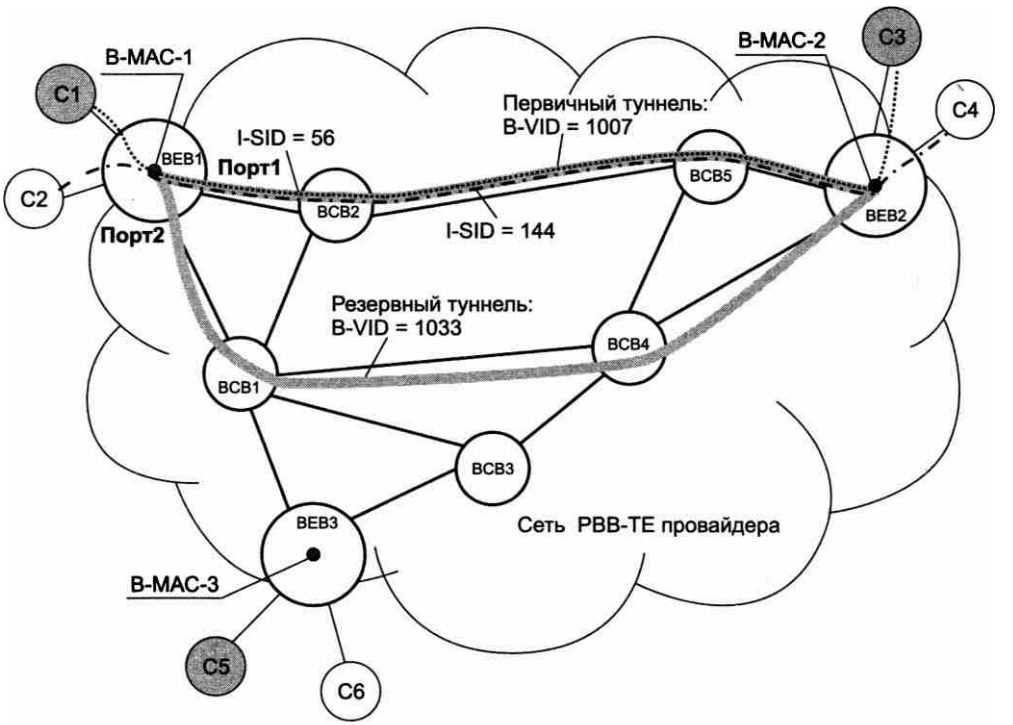


Рис. 21.7. Организация услуг в сети PBB TE

В этой сети сконфигурировано два туннеля:

- основной туннель с B-VID 1007 между BEB1 и BEB2, проходящий через BCB2 и BCB5. Нужно отметить, что, в отличие от туннелей MPLS, туннели PBB TE являются двунаправленными;

- резервный туннель с B-VID 1033, соединяющий те же конечные точки BEB1 и BEB2, но проходящий через другие промежуточные коммутаторы BCB1 и BCB4, что позволяет обеспечить работоспособность резервного туннеля при отказе какого-либо элемента (коммутатора или линии связи) основного туннеля.

Организация обоих туннелей достигается путем ручного конфигурирования таблиц продвижения на всех коммутаторах сети, через которые проходят туннели. Например, таблица продвижения коммутатора BEB1 после такого конфигурирования выглядит как табл. 21.2.

Таблица 21.2. Таблица продвижения коммутатора BEB1

MAC-адрес назначения (B-MAC-DA)	VLAN ID (B-VID)	Выходной порт
B-MAC-2	1007	Port 1
B-MAC-2	1033	Port 2

Для устойчивой работы сети PBB TE необходимо, чтобы комбинация B-VID/B-MAC-DA была уникальной в пределах этой сети.

Как и в технологии PBB, для идентификации магистральных коммутаторов BEB и BCB в технологии PBB-TE используются MAC-адреса обратной связи, которые относятся не к отдельному физическому интерфейсу, а к коммутатору в целом. При ручном задании MAC-адресов ответственность за их уникальность лежит на администраторе; понятно, что такое решение может работать только в пределах одного административного домена.

Добавление значения B-VID к адресу B-MAC-DA позволяет организовать к одному и тому же пограничному коммутатору до 1024 туннелей с различными в общем случае путями прохождения через сеть. Это дает администратору или системе управления широкие возможности в отношении инжиниринга трафика в сетях PBB TE.

Таблицы продвижения в сети PBB TE имеют стандартный вид для коммутаторов, поддерживающих технику VLAN. Изменяется только способ построения этих таблиц — вместо автоматического построения на основе изучения адресов передаваемых кадров имеет место их внешнее формирование.

Отображение пользовательского трафика на соединения I-SID и связывание этих соединений с туннелями B-VID происходит в технологии PBB TE точно так же, как в технологии PBB.

Так как сети PBB TE поддерживают *только двухточечные соединения*, пограничным коммутаторам не нужно изучать пользовательские MAC-адреса.

Отказоустойчивость туннелей PBB TE обеспечивается механизмом, аналогичным рассмотренному ранее в главе 20 механизму защиты пути в технологии MPLS.

Если администратор сети хочет защитить некоторый туннель, он должен сконфигурировать для него резервный туннель и постараться проложить его через элементы сети, не лежащие на пути основного туннеля. В случае отказа первичного туннеля его трафик автоматически направляется пограничным коммутатором в резервный туннель. В примере, приведенном на рис. 21.6, для первичного туннеля с идентификатором B-VID, равным

1007, сконфигурирован резервный туннель с идентификатором B-VID, равным 1033. При отказе туннеля 1007 трафик соединений с идентификаторами I-SID, равными 56 и 144, будет направлен коммутатором BEB1 в туннель 1033.

Для мониторинга состояний первичного и резервного туннелей в технологии PBB TE применяется протокол CFM. Этот протокол является обязательным элементом технологии PBB TE. Мониторинг выполняется путем периодической отправки сообщений CCM каждым пограничным коммутатором туннеля. Время реакции механизма защиты туннелей PBB TE определяется периодом следования сообщений CCM; при аппаратной реализации этого протокола портами коммутатора время реакции может находиться в пределах десятка миллисекунд, то есть соизмеримо с реакцией сетей SDH.

Выводы

Движущими силами превращения Ethernet в транспортную технологию операторского класса CET являются:

- привлекательность для пользователей услуг Ethernet в глобальном масштабе;
- низкая стоимость оборудования Ethernet;
- унификация технологий канального уровня.

Для реализации технологии CET комитет IEE802.1 разработал три стандарта:

- на мосты провайдера (PB);
- на магистральные мосты провайдера (PBB);
- на магистральные мосты провайдера с поддержкой инжиниринга трафика (PBB-TE).

В стандарте на PB виртуальные локальные сети (VLAN) провайдера и пользователей разделены.

В стандарте на PBB разделены как виртуальные локальные сети (VLAN), так и MAC-адреса провайдера и пользователей.

Стандарт на PBB поддерживает только услуги двухточечных соединений, но дает администратору сети полный контроль над путями следования трафика через сеть. Еще одним важным новым свойством этого стандарта является механизм быстрой защиты пользовательских соединений.

Стандарт PBB TE базируется на технологии PBB, но добавляет к ней возможность инжиниринга трафика и механизм быстрого переключения на резервный туннель.

Контрольные вопросы

1. Ethernet операторского класса — это:
   а) внутренняя транспортная технология операторов связи;
   б) новая услуга операторов связи;
   в) маркетинговый термин, обозначающий ту же самую классическую версию Ethernet.
2. Какие улучшения классической версии Ethernet были сделаны для превращения ее в технологию операторского класса? Варианты ответов:

а) повышена производительность оборудования Ethernet;

б) улучшены эксплуатационные свойства оборудования Ethernet;

в) добавлена возможность изоляции адресных пространств клиентов и оператора.

3. С какой целью для сообщений CCM введено понятие уровня? Варианты ответов:

а) для мониторинга иерархических многоуровневых соединений MPLS;

б) для мониторинга многодоменных сетей Ethernet;

в) для обеспечения приоритетности тестирования сети оператора связи.

4. Стандарт «Мосты провайдера» обеспечивает изоляцию:

а) виртуальных локальных сетей клиентов и провайдера;

б) MAC-адресов клиентов и провайдера;

в) MAC-адресов пограничных и магистральных коммутаторов провайдера.

5. Пограничные коммутаторы провайдера, работающие в соответствии со стандартом «Магистральные мосты провайдера», должны изучать MAC-адреса клиентов:

а) всегда;

б) никогда;

в) при оказании услуги E-LAN.

ГЛАВА 22 Виртуальные частные сети

Услуги виртуальных частных сетей

Общие свойства VPN

Сервис **виртуальных частных сетей** (Virtual Private Network, **VPN**) появился как более экономичная альтернатива сервису выделенных каналов, используемому при построении частной компьютерной сети. Каналы виртуальной частной сети, так же как и выделенные каналы, соединяют отдельные сети клиента этой услуги в единую изолированную сеть. Однако в отличие от выделенных каналов, которые строятся с помощью техники коммутации каналов и обладают фиксированной пропускной способностью, каналы виртуальной частной сети прокладываются внутри сети с коммутацией пакетов, такой как IP, MPLS или Ethernet. Экономичность сервиса VPN является следствием более эффективного разделения ресурсов сети при коммутации пакетов по сравнению с коммутацией каналов, реализуемой в рамках построения частной сети.

На рис. 22.1 показан пример построения корпоративной сети клиента А с помощью сервиса виртуальной частной сети; каналы представляют собой соединения в сетях с коммутацией пакетов операторов 1 и 2.

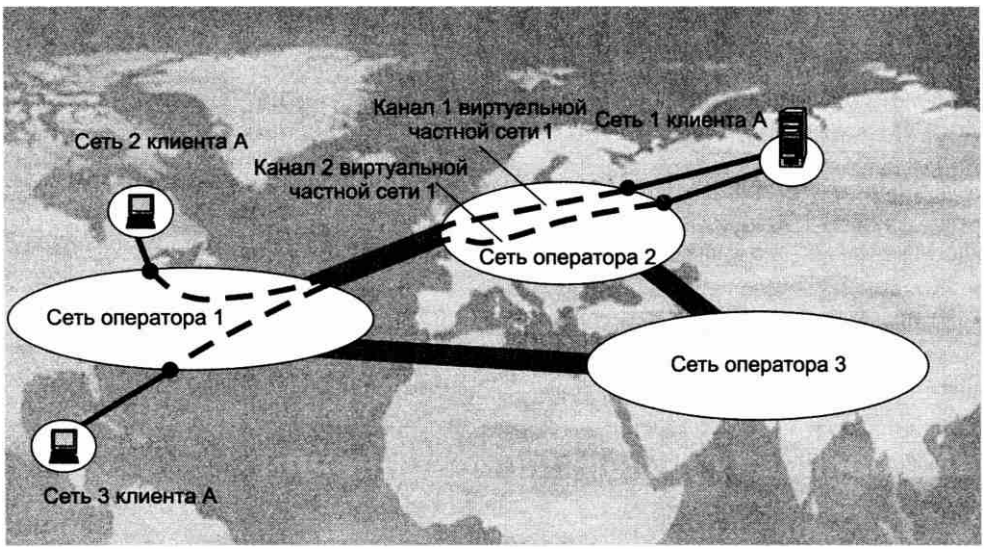


Рис. 22.1. Сервис виртуальной частной сети

Технология VPN позволяет реализовать сервисы, *приближающиеся к сервисам изолированной частной сети по качеству*, но на разделяемой между пользователями инфраструктуре публичной сети с коммутацией пакетов.

Объединяемые с помощью услуги VPN сети пользователя называют также **сайтами**.

Можно сказать, что виртуальная частная сеть имитирует некоторые свойства частной сети, проистекающие из ее изолированности, используя для этого другие технологии. Наиболее ценными для владельцев частных сетей являются следующие их свойства:

- *Ограничение доступа к сети на уровне транспорта*: только узлы сети имеют техническую возможность посылать свои пакеты друг другу. Для технологии VPN обеспечить это свойство очень трудно, так как пакеты пользователей VPN проходят через те же коммуникационные устройства и каналы, что и пакеты внешних пользователей.
- *Независимая система адресации*. В частных сетях нет ограничений на выбор адресов — они могут быть любыми. Чтобы сохранить это свойство, сеть VPN должна допускать адресацию узлов из всего диапазона IP-адресов, включая частные IP-адреса (рекомендованные только для автономного использования).
- *Предсказуемая производительность*. Собственные линии связи гарантируют заранее известную пропускную способность между узлами предприятия (для глобальных соединений) или коммуникационными устройствами (для локальных соединений). Обеспечение предсказуемой пропускной способности в публичной сети с коммутацией пакетов может стать проблемой для сервиса VPN.
- *Максимально возможная безопасность*. Отсутствие связей с внешним миром ограждает частную сеть от атак извне и существенно снижает вероятность «прослушивания» трафика по пути следования пакетов. VPN ограничивает доступ внешних пользователей, а значит, исключает возможность атак извне, а для защиты от прослушивания можно применить шифрование.

Различные технологии VPN отличаются набором свойств частной сети, которые они имитируют, а также степенью приближения к качеству этих свойств.

В зависимости от того, кто реализует услугу VPN, провайдер или клиент, они подразделяются на два вида.

В **поддерживаемой клиентом виртуальной частной сети** (Customer Provided Virtual Private Network, CPVPN) все тяготы по поддержке сети VPN ложатся на плечи потребителя. Провайдер предоставляет только «простые» традиционные услуги общедоступной сети по объединению узлов клиента, например доступ в Интернет, а специалисты предприятия самостоятельно конфигурируют средства VPN и управляют ими.

В случае **поддерживаемой провайдером виртуальной частной сети** (Provider Provisioned Virtual Private Network, PPVPN) провайдер услуг VPN на основе собственной сети воспроизводит частную сеть для каждого своего клиента, изолируя и защищая ее от остальных.

Поддерживаемые провайдером сети VPN обычно обеспечивают более широкий спектр имитируемых свойств частной сети, чем поддерживаемые клиентом. Это объясняется тем, что провайдер имеет контроль над собственной сетью и может применить в ней соответствующую технологию и сконфигурировать свои устройства наиболее эффективным для оказания услуг VPN способом. Клиент такой возможности лишен, он может использовать

стандартный транспортный сервис провайдера и придать ему свойства VPN благодаря специальной конфигурации своих пограничных устройств. Как правило, поддерживаемые клиентом сети VPN используют шифрование трафика и его туннелирование через Интернет, — мы будем рассматривать этот тип сетей VPN в части VII, посвященной безопасности сетей.

В зависимости от того, адресная информация какого уровня принимается во внимание при объединении сетей клиентов, различаются:

- **VPN второго уровня**: учитывается адресная информация второго (канального уровня) сетей клиентов, то есть MAC-адреса и идентификаторы VLAN;
- **VPN третьего уровня**: учитываются IP-адреса сетей клиентов.

Провайдеры для оказания услуг VPN используют различные технологии, они рассматриваются далее в этой главе.

Стандартизация услуг VPN второго уровня

Мы уже подчеркивали значение предоставления пользователям услуг VPN с привычным для пользователей интерфейсом Ethernet. Такие услуги являются доминирующими в группе услуг VPN второго уровня. Внутренняя реализация VPN услуг второго уровня может быть разной, сегодня провайдеры чаще всего используют в этих целях такие технологии, как Carrier Ethernet (то есть технологии PB, PBB и PBB-TE) и MPLS. Эти две популярные реализации услуг VPN второго уровня получили названия Ethernet over Ethernet (EoE) и Ethernet over MPLS (EoMPLS). Наличие у этих услуг интерфейса Ethernet дает им еще одно название — Ethernet VPN.

Очень часто при описании характеристик услуги VPN (например, топологии связей между пользовательскими сетями) провайдер применяет собственную терминологию, при этом описание может быть технологически ориентировано: например, услуги EoMPLS могут быть описаны с привлечением терминологии MPLS.

Поэтому понятно, что стандартизация услуг Ethernet VPN — это важное направление работ в области Ethernet операторского класса, позволяющее провайдерам и пользователям однозначно описывать услуги, не вдаваясь в детали их внутренней реализации.

Работой по созданию технологически нейтральных спецификаций глобальной услуги Ethernet VPN Ethernet занимается организация под названием Metro Ethernet Forum (MEF).

В спецификациях MEF вводится три типа услуг виртуальных частных сетей Ethernet, которые отличаются *топологией* связей между сайтами пользователей. Для того чтобы формализовать топологию связей, вводится понятие **виртуального соединения Ethernet** (Ethernet Virtual Circuit, EVC). Каждое соединение EVC связывает сайты пользователей в отельную виртуальную частную сеть, объединяя сетевые интерфейсы пользователей (UNI).

Соответственно имеются три типа соединений EVC (рис. 22.2):

- «точка-точка» (двухточечная топология);
- «каждый с каждым» (полносвязная топология);
- «дерево» (древовидная топология).

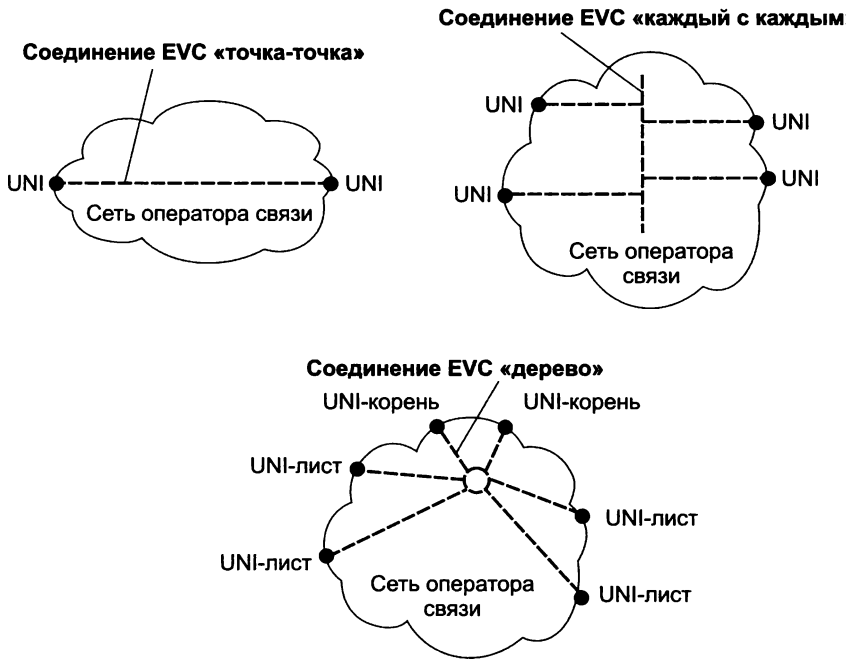


Рис. 22.2. Три типа услуг Ethernet

В зависимости от типа используемого соединения различаются и типы услуг:

- **E-LINE**. Эта услуга связывает только два пользовательских сайта через двухточечное EVC-соединение. Услуга E-LINE соответствует услуге выделенной линии.
- **E-LAN**. Эта услуга аналогична услуге локальной сети, так как она позволяет связать неограниченное число пользовательских сайтов таким образом, что каждый сайт может взаимодействовать с каждым. При этом соблюдается логика работы локальной сети — кадры Ethernet с неизученными и широковещательными MAC-адресами передаются всем сайтам, а кадры с изученными уникальными MAC-адресами — только тому сайту, в котором находится конечный узел с данным адресом.
- **E-TREE**. Спецификация этой услуги появилась позже других; в локальных сетях ей аналога нет. Пользовательские сайты делятся на корневые и листовые. Листовые сайты могут взаимодействовать только с корневыми, но не между собой. Корневые сайты могут взаимодействовать с листовыми сайтами и друг с другом.

Кроме того, в спецификациях MEF вводятся два варианта каждого типа услуги. В первом варианте пользовательский сайт определяется как сеть, подключенная к отдельному физическому интерфейсу UNI. Значения идентификаторов VLAN в пользовательских кадрах (то есть значения C-VID в терминологии PB/PBB/PBB-TE) в расчет не принимаются. В названии этого варианта услуги к названию типа добавляется термин «частный» (private): например, для услуги типа E-LINE этот вариант называют частной линией Ethernet (Ethernet Private Line, **EPL**), а для услуги E-LAN — частной локальной сетью Ethernet (Ethernet Private LAN, **EPLAN**).

В другом варианте услуги к одному и тому же физическому интерфейсу UNI могут быть подключены различные пользовательские сайты. В этом случае они различаются по значению идентификатора VLAN (C-VID). Другими словами, провайдер внутри своей сети сохраняет деление локальной сети на VLAN, сделанное пользователем. В варианте услуги с учетом VLAN добавляется название «виртуальная частная»: например, для услуги типа E-LINE это будет виртуальная частная линия Ethernet (Ethernet Virtual Private Line, **EVPL**), а для услуги E-LAN — виртуальная частная локальная сеть Ethernet (Ethernet Private LAN, **EVPLAN**).

В своих определениях MEF использует термины «частная услуга» и «виртуальная частная услуга» не совсем традиционным образом, так как оба типа услуги являются виртуальными частными в том смысле, что они предоставляются через логическое соединение в сети с коммутацией пакетов, а не через физический канал в сети с коммутацией каналов.

Помимо указанных определений услуг спецификации MEF стандартизуют некоторые важные параметры услуг: например, услуга может характеризоваться гарантированным уровнем пропускной способности соединения, а также гарантированными параметрами QoS.

Технологии Carrier Ethernet Transport (PB, PBB и PBB-TE) оказывают услуги Ethernet VPN непосредственно, без дополнительных надстроек и механизмов, они и разрабатывались для этой цели. Сети PB и PBB могут оказывать услуги E-LINE (при соединении двух пользовательских сайтов) и E-LAN (при соединении более чем двух пользовательских сайтов), а сети PBB-TE — только услуги E-LINE. Услуги E-TREE ни одна из этих технологий не поддерживает.

Технология MPLS VPN второго уровня

Для использования MPLS как внутренней технологии провайдера при предоставлении услуг Ethernet VPN маршрутизаторы MPLS должны быть сконфигурированы специальным образом, а пограничные маршрутизаторы должны, кроме того, предоставлять пользователям интерфейсы Ethernet.

Псевдоканалы

Стандарты IETF описывают два типа услуг Ethernet VPN, которые строятся с помощью технологии MPLS: **VPWS** (Virtual Private Wire Service) и **VPLS** (Virtual Private LAN Service). Различие между этими услугами в том, что VPWS эмулирует соединение Ethernet с двухточечной топологией, то есть канал Ethernet, а VPLS эмулирует поведение локальной сети, то есть обеспечивает соединения с полносвязной топологией в стиле обычной локальной сети Ethernet.

Если использовать терминологию MEF, то услуга VPLS соответствует услуге E-LAN, а услуга VPWS — услуге E-LINE. При этом стандарты IETF описывают оба варианта услуг как приннимая во внимание пользовательские идентификаторы VLAN, то есть услуги EVPL и EVPLAN, так и нет, то есть EPL и EPLAN.

Обе услуги являются услугами MPLS VPN второго уровня (MPLS L2VPN), так как они позволяют предоставлять услуги VPN, взаимодействуя с пользовательскими сетями на втором уровне.

Основным строительным элементом этих услуг являются так называемые **псевдоканалы**[1] (pseudowire), которые соединяют пограничные маршрутизаторы провайдера.

На рис. 22.3 показано три таких псевдоканала, соединяющих между собой пограничные маршрутизаторы PE1–PE4 (PE — от Provider Edge).

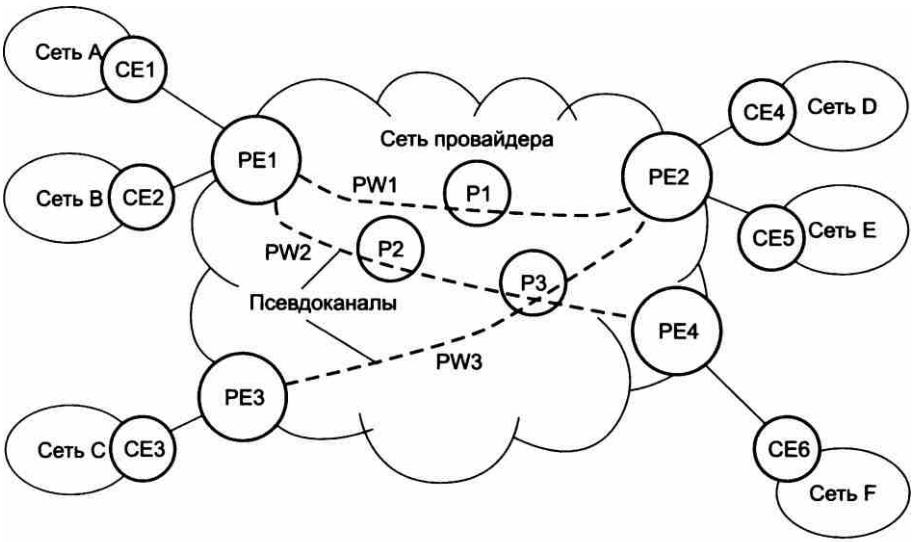


Рис. 22.3. Псевдоканалы в сети провайдера

Псевдоканалы представляют собой пути LSP второго уровня иерархии (называемого также внутренним уровнем), проложенные внутри LSP первого (внешнего) уровня. Обычно в качестве LSP первого уровня иерархии используются TE-туннели MPLS, так как они обладают полезными дополнительными свойствами, которых нет у путей, проложенных с помощью протокола LDP, например более сбалансированной нагрузкой. На рисунке пути LSP первого уровня не показаны, чтобы заострить внимание читателя на псевдоканалах.

Псевдоканалы — это логические транспортные соединения, физически они могут проходить через промежуточные магистральные маршрутизаторы, однако для них они прозрачны, то есть в нашем примере маршрутизаторы P1, P2 и P3 просто не замечают их существования в сети.

Однако псевдоканал — это не просто логическое соединение LSP второго уровня иерархии согласно определению, данному в RFC 3985, у псевдоканала есть более специфическое назначение.

> Псевдоканал — это механизм, который эмулирует существенные свойства какого-либо телекоммуникационного сервиса через сеть с коммутацией пакетов.

[1] Встречаются и другие русские переводы термина pseudowire, например эмулятор канала, эмулятор кабеля, псевдопровод.

Одним из вариантов применения псевдоканалов при эмуляции услуг Ethernet является передача псевдоканалом трафика одного пользовательского соединения, при этом псевдоканал эмулирует кабельное соединение между сетями пользователей. В примере на рисунке псевдоканал PW2 служит для организации соединения между сетями A и F через сеть провайдера. При этом кадры Ethernet, отправляемые сетью A в сеть F, инкапсулируются пограничным маршрутизатором PE1 в данные псевдоканала и доставляются им пограничному маршрутизатору PE2, который извлекает эти кадры и отправляет их в сеть F в первоначальном виде.

Из определения, данного в RFC 3985, видно, что назначение псевдоканала шире эмуляции Ethernet, — это может быть и эмуляция сервисов выделенных каналов технологий PDH или SDH, и эмуляция виртуальных каналов ATM или Frame Relay; однако в любом случае эмуляция такой услуги выполняется через *пакетную сеть*. Тип пакетной сети также не уточняется, так что это может быть и классическая сеть IP (без MPLS), и сеть IP/MPLS, и сеть ATM. Главное в этом обобщенном определении то, что псевдоканал скрывает от пользователей эмулируемого сервиса детали пакетной сети провайдера, соединяя пользовательские пограничные устройства (CE на рис. 22.3) таким образом, как если бы они соединялись с помощью выделенного канала или кабеля.

Для некоторых наиболее важных сочетаний эмулируемого сервиса и типа пакетной сети комитет IETF разработал отдельные спецификации псевдоканалов. Далее мы рассмотрим только один тип псевдоканала, который нужен для предоставления услуг Ethernet операторского класса, а именно *псевдоканал эмуляции Ethernet* через сети IP/MPLS, описанный в RFC 4448.

Технически создать LSP второго уровня достаточно просто — для этого в маршрутизаторах, соединенных LSP первого уровня, нужно задать значение метки второго уровня, которое будет использоваться, чтобы различать LSP второго уровня внутри LSP первого уровня. Этот процесс иллюстрирует рис. 22.4. На нем изображены два пограничных маршрутизатора PE1 и PE2, соединенные псевдоканалом PE57. Однако рисунок оказался немного сложнее, чем можно было предположить, — вместо одного пути LSP первого уровня мы видим два таких пути. Это связано с тем, что двухточечные псевдоканалы, которые служат для эмуляции Ethernet, по определению IETF всегда являются двунаправленными[1], а MPLS LSP — это однонаправленный путь. Поэтому для создания двунаправленного псевдоканала требуется два однонаправленных пути второго уровня, вложенных в два однонаправленных пути первого уровня, что и показано на рисунке.

Рассматриваемый в нашем примере псевдоканал в направлении от PE1 к PE2 идентифицируется меткой 57, а туннель, который использует этот канал, — меткой 102. Поэтому при отправке кадра Ethernet, предназначенного для PE2, маршрутизатор PE1 помещает исходный кадр Ethernet в кадр MPLS и адресует этот кадр двумя метками: внешней меткой 102 и внутренней меткой 57. Внешняя метка применяется затем магистральными маршрутизаторами P1, P2 и P3 для того, чтобы доставить кадр пограничному маршрутизатору PE2, при этом в процессе передачи кадра происходит обычная коммутация по меткам (на рисунке показано, что после прохождения P1 внешняя метка получила значение 161). Внутренняя метка 57 требуется только пограничному маршрутизатору PE2, который знает,

---

[1] Форум IETF определил и другие типы псевдоканалов, такие как «точка-многоточка» и «многоточка-многоточка». Эти псевдоканалы являются однонаправленными, но для эмуляции Ethernet они не используются.

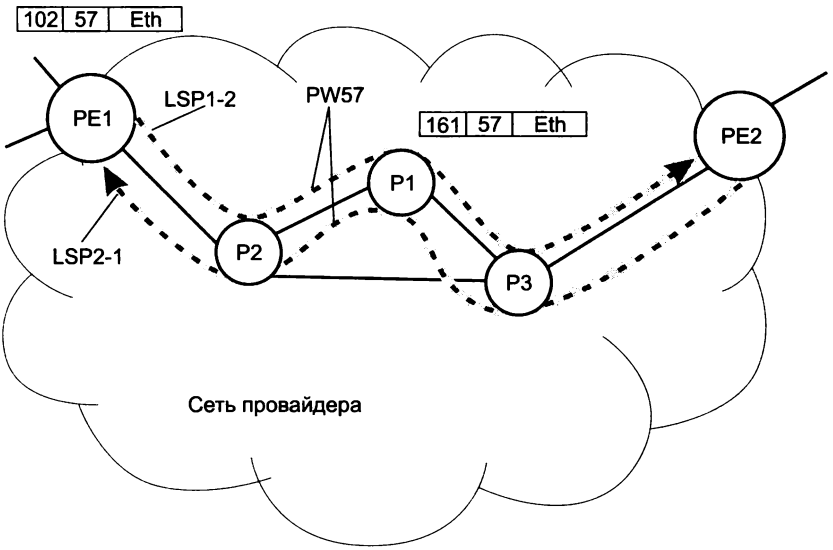


Рис. 22.4. Создание псевдоканала внутри туннелей MPLS

что эта метка соответствует псевдоканалу PW57, необходимому для связи с некоторой пользовательской сетью.

Как мы видим из рассмотренного примера, псевдоканалы работают только внутри сети провайдера, так что для эмуляции сервиса «из конца в конец» нужны еще какие-то элементы и механизмы, — и мы скоро их рассмотрим, но сначала давайте обсудим преимущества применения псевдоканалов поверх MPLS. Возникает естественный вопрос: нельзя ли обойтись LSP первого уровня для передачи трафика Ethernet через сеть провайдера, не используя концепцию псевдоканала? В принципе, без псевдоканалов обойтись можно, но тогда для каждого нового пользовательского соединения пришлось бы создавать новый туннель (то есть LSP первого уровня), а это не очень масштабируемое решение, так как конфигурирование такого пути обязательно подразумевает конфигурирование всех магистральных маршрутизаторов сети. Поэтому одно из существенных преимуществ псевдоканалов состоит в том, что в сети провайдера нужно сконфигурировать только сравнительно небольшое число туннелей между пограничными маршрутизаторами, а затем использовать каждый из них для прокладки необходимого числа псевдоканалов. Создание нового псевдоканала также требует конфигурирования, но только пары пограничных маршрутизаторов, которые являются конечными точками псевдоканала, а это подразумевает гораздо меньший объем работы. Мы уже видели применение подобной схемы в сетях PBB и PBB-TE, где роль псевдоканалов играют соединения I-SID.

Другим преимуществом псевдоканалов является их универсальность, то есть возможность их применения не только в сетях MPLS, но и в сетях других типов, таких как «чистые» IP-сети с туннелированием (например, по протоколу L2TP или GRE) и не только при эмуляции Ethernet, но и при эмуляции других сервисов (например, каналов PDH). Естественно, что при переходе к другой реализации псевдоканалов конкретные команды конфигурирования меняются, но концепция остается, и это помогает администраторам сети освоить новую технологию.

Услуги VPWS

Услуги **виртуальных частных каналов** (Virtual Private Wire Service, **VPWS**) исполняют роль «глобального кабеля», соединяя прозрачным образом две локальные пользовательские сети Ethernet через сеть оператора связи. Мы рассмотрим организацию такой услуги с помощью псевдоканалов MPLS на примере (рис. 22.5). При этом мы опишем дополнительные элементы механизма эмуляции услуги Ethernet, которые были опущены при описании псевдоканалов.

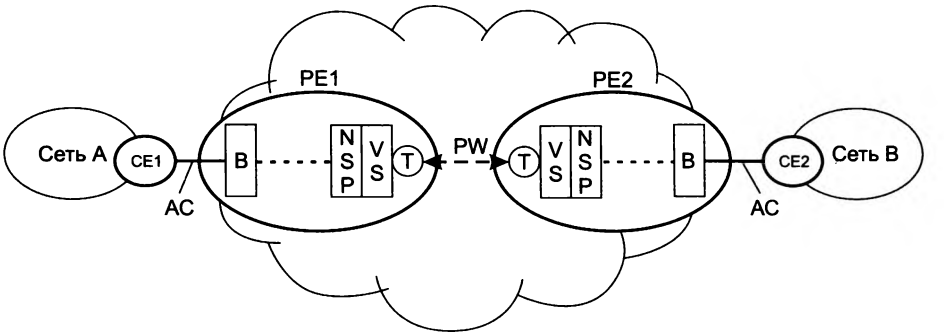


Рис. 22.5. Организация виртуального частного канала Ethernet

Чаще всего пользовательские сети соединяются с пограничным маршрутизатором провайдера через выделенный интерфейс, который для глобальных услуг Ethernet должен быть стандартным интерфейсом Ethernet, например 1000Base-LX. В этом случае услуга VPWS заключается в прозрачном соединении этих интерфейсов, когда сеть провайдера передает все кадры, которые поступают на такой интерфейс от сети пользователя. Иногда этот режим VPWS называют коммутацией портов пользователя. В терминологии MEF это услуга EPL.

Возможен и другой вариант услуги VPWS, когда сеть провайдера соединяет виртуальные пользовательские сети, то есть по двухточечному соединению передаются не все кадры, поступающие через интерфейс пользователя, а только кадры, принадлежащие определенной сети VLAN. Этот режим работы VPWS можно назвать коммутацией виртуальных локальных сетей, или VLAN-коммутацией. В терминологии MEF это услуга EVPL.

Для того чтобы обобщить понятие интерфейса с пользователем, форум IETF ввел термин **канал присоединения** (Attachment Circuit, AC). AC поставляет входной поток пользовательских данных для сети провайдера, то есть ту нагрузку, которую нужно коммутировать. Употребляя этот термин, можно сказать, что услуга VPWS всегда соединяет два пользовательских канала присоединения; такое определение справедливо не только для услуг Ethernet, но и для услуг, например, Frame Relay или ATM, в этом случае каналы присоединения являются виртуальными каналами этих технологий.

На рисунке показаны также внутренние функциональные элементы пограничных маршрутизаторов PE1 и PE2, которые эмулируют услуги VPWS вместе с псевдоканалом PW.

Модуль B (от Bridge — мост) работает по стандартному алгоритму IEEE 802.1D. Его роль в схеме эмуляции — выделение кадров Ethernet из общих потоков, поступающих на порты маршрутизатора, для передачи в псевдоканал. Тем самым модуль моста формирует логи-

ческий интерфейс виртуального коммутатора. Например, если это режим коммутации портов, то модуль моста конфигурируется так, чтобы все кадры, пришедшие на соответствующий порт от пользователя, направлялись для дальнейшей обработки в псевдоканал. Если же это VLAN-коммутация, то модуль моста выбирает для передачи псевдоканалу только кадры, помеченные определенным значением тега VLAN.

Выбранные модулем моста кадры поступают в псевдоканал не непосредственно, а через два промежуточных модуля — NSP и VS. Модуль NSP (Native Service Processing) обеспечивает предварительную обработку кадров Ethernet. Чаще всего такая обработка связана с изменением или добавлением тега VLAN, что может потребоваться, например, если объединяемые пользовательские сети применяют различные значения VLAN для одной и той же виртуальной сети. Модуль VS (Virtual Switch — виртуальный коммутатор) коммутирует один из каналов присоединения с одним из псевдоканалов. Для услуги VPWS этот модуль работает «вхолостую», выполняя постоянную коммутацию единственного канала присоединения с единственным псевдоканалом. Однако для услуги VPLS, которая рассматривается в следующем разделе, виртуальный коммутатор играет важную роль, поэтому в обобщенной схеме эмуляции услуг Ethernet, представленной на рис. 22.5, он присутствует.

После обработки пришедшего кадра модулями NCP и VS он передается псевдоканалу. Конечные точки T псевдоканала PW57 выполняют две операции:

- инкапсуляцию и декапсуляцию пользовательских кадров в кадры MPLS;
- мультиплексирование и демультиплексирование псевдоканалов в туннеле MPLS.

Процедуру инкапсуляции и формат результирующего кадра определяет спецификация RFC 4448. У исходного кадра отбрасываются поля преамбулы и контрольной суммы, после чего он помещается в кадр MPLS с двумя полями меток: внешней (метка туннеля) и внутренней (метка псевдоканала), как это показано на рис. 22.6. На рисунке не показаны поля заголовка кадра MPLS, относящиеся к конкретной канальной технологии, которая используется на внутренних интерфейсах пограничных маршрутизаторов, — как вы помните, кадры MPLS могут иметь обрамление Ethernet, PPP, ATM или Frame Relay (в случае Ethernet это обрамление не имеет отношения к пользовательскому кадру Ethernet, инкапсулированному в кадр MPLS).

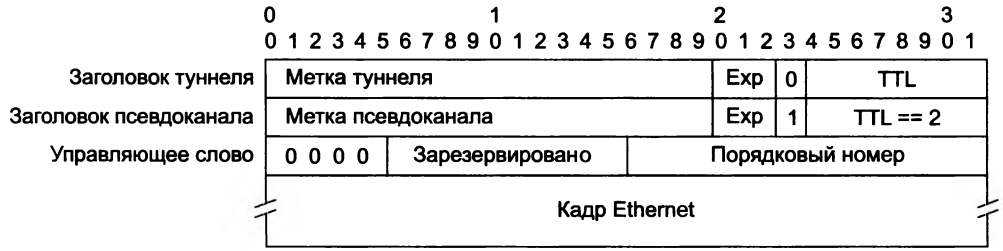


Рис. 22.6. Формат инкапсуляции Ethernet поверх MPLS (RFC 4448)

В то время как первые два слова в заголовке, представленном на рисунке, являются стандартными заголовками MPLS, третье слово, называемое управляющим (control word), впервые появилось в стандарте RFC 4448. Оно не обязательно и предназначено для упорядочивания кадров, передаваемых по псевдоканалу, — для этого каждому кадру

маршрутизатором-отправителем присваивается порядковый номер, который помещается в управляющее слово. Потребность в управляющем слове возникает тогда, когда внутри сети провайдера происходит распараллеливание трафика туннеля и кадры могут выходить из туннеля не в том порядке, в котором были посланы.

Конфигурирование псевдоканалов, то есть согласование внутренних меток, используемых для идентификации и мультиплексирования псевдоканалов внутри туннеля, может быть автоматизировано. Для этого сегодня применяют протокол LDP или BGP. Обратите внимание, что речь идет о прокладке псевдоканала, а не самого туннеля, эти два процесса независимы, так что туннель может быть проложен, например, с помощью протокола RSVP TE, а псевдоканалы в нем — с помощью протокола LDP.

Протокол LDP служит также для уведомления одним маршрутизатором PE другого об изменении состояния «работоспособен—неработоспособен» псевдоканала или канала присоединения. Это очень полезное свойство, так как без него удаленный маршрутизатор PE не узнает об отказе непосредственно не присоединенных к нему отрезков эмулируемого транспортного соединения и будет пытаться его использовать, посылая данные. Протокол LDP позволяет в случае такого отказа отозвать метку, ранее назначенную псевдоканалу.

В завершение описания услуг VPWS хочется напомнить, что такое важное свойство услуги, как гарантированная пропускная способность, обеспечивается с помощью техники инжиниринга трафика, опирающейся в данном случае на соответствующие свойства туннелей MPLS. Аналогично обстоит дело с параметрами качества обслуживания (QoS) для виртуальных соединений VPWS — они могут быть обеспечены с помощью стандартных механизмов QoS, таких как, например, приоритетное обслуживание, профилирование трафика, контроль доступа и резервирование ресурсов. И в этом случае MPLS является хорошим базисом, так как детерминированность маршрутов туннелей MPLS делает контроль доступа намного более определенной процедурой, чем в случае IP-сетей с их распределенным (и вносящим неопределенность) механизмом выбора маршрутов.

Услуги VPLS

Услуги **виртуальной частной локальной сети** (Virtual Private LAN Service, **VPLS**) соответствуют определению услуг E-LAN MEF, причем как варианту с учетом идентификаторов VLAN пользователей EVPLAN, так и варианту без их учета EPLAN.

Так же как и в случае VPWS, сервис VPLS организован на базе псевдоканалов. Отличие заключается в том, что для каждого экземпляра VPLS используется *отдельный набор псевдоканалов*. При этом каждый такой набор имеет полносвязную топологию, то есть все пограничные маршрутизаторы PE, участвующие в работе какого-то экземпляра VPLS, связаны друг с другом.

На рис. 22.7 показан пример сети провайдера, эмулирующей два сервиса VPLS. Пользовательские сети C1, C5 и C8 относятся к «серому» сервису VPLS, а сети C2, C3, C4, C6 и C7 — к «белому». Соответственно набор псевдоканалов PW-B1, PW-B2 и PW-B3 объединяет пограничные маршрутизаторы, к которым подключены сети «серого» сервиса VPLS, а набор псевдоканалов PW-W1, PW-W2 и PW-W3 — маршрутизаторы, к которым подключены сети «белого» сервиса VPLS (в нашем примере это одни и те же пограничные маршрутизаторы PE1, PE2 и PE3, но если бы сети C4 не существовало, то псевдоканалы PW-W2 и PW-W3 были бы не нужны).

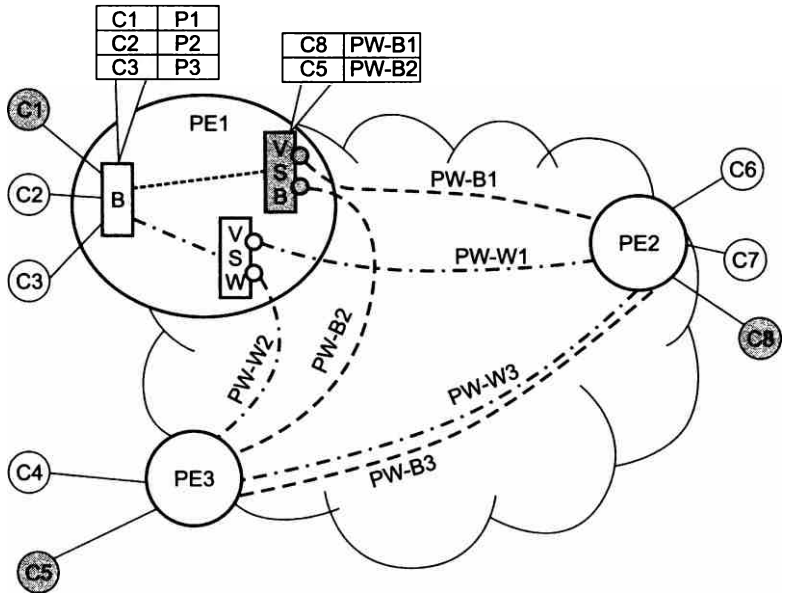


Рис. 22.7. Организация услуги VPLS

Внутренняя организация пограничного маршрутизатора при оказании услуги VPLS показана на примере маршрутизатора PE1. Мы видим, что для поддержки каждого экземпляра сервиса VPLS пограничному маршрутизатору требуется *отдельный виртуальный коммутатор*, в данном случае это модули VSB и VSW (модули NSP не показаны, чтобы не загромождать рисунок, но они в PE1 входят, по одному на каждый экземпляр VPLS).

Как и в случае VPWS, модуль B выполняет стандартные функции моста и при этом формирует логический интерфейс с каждым из виртуальных коммутаторов. Этот интерфейс может также формироваться на основе коммутации либо пользовательских портов, когда весь трафик от определенного порта (или нескольких портов) передается на логический интерфейс, либо сетей VLAN, когда выбираются кадры одной (или нескольких) пользовательских сетей VLAN от одного или нескольких портов.

Однако если в случае VPWS виртуальный коммутатор выполнял простую работу по передаче кадров от логического интерфейса, то для VPLS этот модуль функционирует по алгоритму стандартного коммутатора (моста). Для этого виртуальный коммутатор изучает MAC-адреса и строит свою таблицу продвижения, как и обычный коммутатор. На рисунке показан упрощенный вид таблицы продвижения PE1, состоящей из двух записей: одна запись связывает адрес M8 сети C8 с псевдоканалом PW-B1, другая — адрес M5 сети C5 с псевдоканалом PW-B2. Пользуясь такой таблицей, виртуальный коммутатор не затапливает сеть, получая кадры с адресами M5 или M8, а направляет их в псевдоканал, ведущий к пограничному коммутатору, к которому подключена сеть с узлом назначения. Кадры с широковещательным адресом или адресом, отсутствующим в таблице продвижения, поступают на все его псевдоканалы, в данном случае — на PW-B1 и PW-W1.

> Единственной особенностью виртуального коммутатора является то, что он не изучает адреса отправления кадров, приходящих с логического интерфейса.

Эта операция не нужна, потому что для интерфейсов, представленных псевдоканалами, виртуальный коммутатор работает по *правилу расщепления горизонта* (split horizon) — он никогда не передает на псевдоканал кадры, полученные от какого бы то ни было псевдоканала. Тем самым предотвращается образование петель между виртуальными коммутаторами, а доставку кадров по назначению гарантирует полносвязная топология. То есть любой кадр, полученный виртуальным коммутатором по псевдоканалу, всегда передается на логический интерфейс пользователя, соответствующий тому сервису VPLS, к которому относится псевдоканал.

Модуль моста B изучает только адреса, приходящие с пользовательских интерфейсов. Они служат ему для выбора нужного интерфейса в том случае, когда несколько пользовательских сетей относятся к одному сервису VPLS.

Конфигурирование PE может *оказаться* трудоемким занятием, так как в случае N пограничных коммутаторов нужно создать $N(N-1)/2$ псевдоканалов. Кроме того, добавление любого нового устройства PE требует переконфигурирования всех остальных коммутаторов. Для автоматизации этих процедур можно использовать вариант организации VPLS, описанный в RFC 4761, так как он предусматривает применение для этой цели протокола BGP. Вариант VPLS, описанный в RFC 4762, подразумевает распределение меток второго уровня иерархии с помощью протокола LDP, автоматизацию процедур конфигурирования он не поддерживает.

Технология MPLS VPN третьего уровня

В этом типе VPN пользовательские сети (называемые также сайтами) объединяются на основе адресной информации третьего уровня, то есть IP-адресов (а не MAC-адресов и идентификаторов VLAN, как в MPLS VPN второго уровня). При этом IP-адреса могут быть как публичными, так и частными, в последнем случае они должны быть уникальными в пределах одной виртуальной сети.

В то же время между MPLS VPN третьего уровня и второго уровня имеется много общего:

- услуги предоставляются провайдером с помощью сети IP/MPLS;
- пограничные маршрутизаторы PE выполняют всю работу по поддержанию VPN;
- внутренние маршрутизаторы провайдера P нужны только для передачи MPLS пакетов между пограничными маршрутизаторами PE; они не знают о существовании VPN;
- для передачи информации о принадлежности пакета к определенной сети VPN используется *метка MPLS второго уровня*.

Разграничение маршрутной информации

Каждый пограничный маршрутизатор PE обменивается маршрутной информацией с соединенными с ним клиентскими маршрутизаторами CE по какому-нибудь протоколу маршрутизации класса IGP, например OSPF или IS-IS (рис. 22.8). С каждым из клиентов может использоваться свой протокол IGP, то есть с сайтом A — протокол OSPF, а с сайтом B — протокол IS-IS. С помощью этих протоколов маршрутизатор узнает о том, какие сети достижимы в сайтах клиентов. Кроме того, маршрутизатор PE поддерживает сеанс протокола IGP с остальными маршрутизаторами сети провайдера (как P, так и PE) для того, чтобы знать топологию этой сети и маршрутизировать пакеты в пределах этой сети.

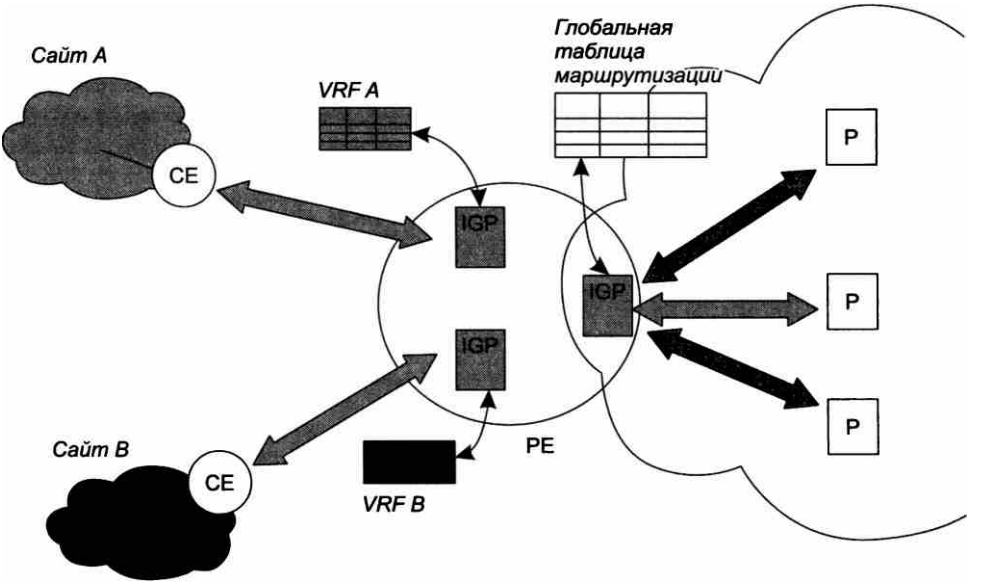


Рис. 22.8. Разграничение маршрутных объявлений в сети MPLS VPN третьего уровня

Для корректной работы VPN требуется, чтобы информация о маршрутах через сеть провайдера не распространялась за ее пределы, а сведения о маршрутах в клиентских сайтах не становились известными за границами определенных сетей VPN.

Барьеры на пути распространения маршрутных объявлений могут устанавливаться соответствующим конфигурированием маршрутизаторов PE. Протоколы маршрутизации этих маршрутизаторов должны быть оповещены о том, с каких интерфейсов и от кого они имеют право принимать объявления определенного сорта и на какие интерфейсы и кому их распространять.

Можно представить, что через маршрутизатор PE проходит невидимая граница между зоной клиентских сайтов и зоной ядра сети провайдера. По одну сторону располагаются интерфейсы, через которые PE взаимодействует с маршрутизаторами P, а по другую — интерфейсы, к которым подключаются сайты клиентов. С одной стороны на PE поступают объявления о маршрутах в сети провайдера, с другой — объявления о маршрутах в сетях клиентов.

На рис. 22.8 показан маршрутизатор PE, на котором установлены несколько протоколов класса IGP. Один из них сконфигурирован для приема и распространения маршрутных объявлений только с тех трех внутренних интерфейсов, которые связывают этот маршрутизатор PE с маршрутизаторами P. Два других протокола IGP обрабатывают маршрутную информацию от сайтов клиентов.

Аналогичным образом настроены и остальные маршрутизаторы PE. Таблица маршрутизации, создаваемая на пограничных маршрутизаторах PE на основе объявлений из магистральной сети провайдера, имеет специальное название: **глобальная таблица маршрутизации**. В ней содержатся маршруты в пределах внутренней сети провайдера,

информации о маршрутах в сетях клиентов в ней нет. Таблицы маршрутизации, которые PE формирует на основе объявлений, поступающих от сайтов клиентов, получили название таблиц **VRF** (VPN Routing and Forwarding). В них имеется только информация о сетях клиентов.

Маршрутизаторы P принимают и обрабатывают маршрутную информацию IGP, поступающую со всех интерфейсов. В создаваемых ими таблицах маршрутизации имеется информация только о сетях провайдера.

Сайты клиентов представляют собой обычные сети IP, маршрутная информация в которых может передаваться и обрабатываться с помощью любого протокола маршрутизации класса IGP. Очевидно, что этот процесс никак не регламентируется провайдером. Маршрутные объявления свободно распространяются между узлами в пределах каждого сайта до тех пор, пока не доходят до пограничных маршрутизаторов PE, служащих преградой для их дальнейшего распространения.

Разграничение маршрутов разных клиентов обеспечивает установка на маршрутизаторах PE *отдельной копии протокола маршрутизации* на каждый интерфейс, к которому подключен сайт клиента. Этот протокол принимает и передает клиентские маршрутные объявления только с одного определенного для него интерфейса, не пересылая их ни на внутренние интерфейсы, через которые PE связан с маршрутизаторами P, ни на интерфейсы, к которым подключены сайты других клиентов.

Несколько упрощая, можно считать, что на каждом маршрутизаторе PE создается столько таблиц VRF, сколько сайтов к нему подключено. Фактически на маршрутизаторе PE организуется несколько виртуальных маршрутизаторов, каждый из которых работает со своей таблицей VRF. Возможно и другое соотношение между сайтами и таблицами VRF. Например, если к некоторому маршрутизатору PE подключено несколько сайтов одной и той же сети VPN, то для них может быть создана общая таблица VRF. На рис. 22.8 показаны две таблицы VRF, одна из которых содержит описание маршрутов к узлам сайта A, а другая — к узлам сайта B. К каждой такой таблице можно получить доступ только с сайтов, относящихся к этой же сети VPN.

Обмен маршрутной информацией

Чтобы связать территориально разнесенные сайты заказчика в единую сеть, необходимо, во-первых, создать для них общее пространство распространения маршрутной информации, а во-вторых, проложить во внутренней сети пути, по которым принадлежащие разным сайтам узлы одной и той же сети VPN могли бы вести обмен данными защищенным образом.

Механизмом, с помощью которого сайты одной сети VPN обмениваются маршрутной информацией, является **многопротокольное расширение для BGP** (MultiProtocol extensions for BGP-4, **MP-BGP**). С помощью этого протокола пограничные маршрутизаторы PE организуют взаимные сеансы и в рамках этих сеансов обмениваются маршрутной информацией из своих таблиц VRF.

Особенность протокола BGP и его расширений заключается в том, что он получает и передает свои маршрутные объявления не всем непосредственно связанным с ним маршрутизаторам, как протоколы IGP, а только тем, которые указаны в конфигурационных параметрах в качестве соседей. Маршрутизаторы PE сконфигурированы так, что все получаемые от

клиентских сайтов маршрутные объявления они с помощью MP-BGP пересылают определенным пограничным маршрутизаторам PE. Вопрос о том, кому отправлять маршрутные объявления, а кому нет, целиком зависит от топологии виртуальных частных сетей, поддерживаемых данным провайдером.

Таким образом, кроме маршрутов, поступающих от непосредственно подсоединенных к PE сайтов, каждая таблица VRF дополняется маршрутами, получаемыми от других сайтов данной сети VPN по протоколу MP-BGP. Целенаправленное распространение маршрутов между маршрутизаторами PE обеспечивается надлежащим выбором атрибутов протокола MP-BGP (эти атрибуты описаны в RFC 4360), что детально рассматривается далее.

Независимость адресных пространств сайтов

Одним из свойств частных сетей является независимость их адресных пространств. MPLS VPN третьего уровня имитируют это свойство, разрешая использовать одно и то же адресное пространство, например пространство частных IP-адресов, во всех экземплярах VPN провайдера. При этом в пределах одной и той же сети VPN адреса не должны повторяться, иначе сайты не смогут взаимодействовать друг с другом.

Использование в разных сетях VPN одного и того же адресного пространства создает проблему для маршрутизаторов PE. Протокол BGP изначально был разработан в предположении, что все адреса, которыми он манипулирует, во-первых, относятся к семейству адресов IPv4, во-вторых, однозначно идентифицируют узлы сети, то есть являются глобально уникальными в пределах всей составной сети. Ориентация на глобальную уникальность адресов выражается в том, что, получив очередное маршрутное объявление, протокол BGP анализирует его, не обращая внимания на то, какой сети VPN принадлежит полученный маршрут. Если на вход BGP поступают описания маршрутов к узлам разных сетей VPN, но с совпадающими адресами IPv4, то BGP считает, что все они ведут к одному и тому же узлу, а следовательно, как и полагается в таком случае, он помещает в соответствующую таблицу VRF только один кратчайший маршрут.

Проблема решается за счет применения вместо потенциально неоднозначных адресов IPv4 расширенных и однозначных адресов нового типа, а именно адресов VPN-IPv4, получаемых в результате преобразования исходных адресов IPv4. Преобразование заключается в том, что ко всем адресам IPv4, составляющим адресное пространство той или иной сети VPN, добавляется префикс, называемый **различителем маршрутов** (Route Distinguisher, **RD**). RD уникально идентифицирует каждую сеть VPN. В результате на маршрутизаторе PE все адреса, относящиеся к разным сетям VPN, обязательно будут отличаться друг от друга, даже если они имеют совпадающую часть — адрес IPv4.

Здесь оказалась полезной способность расширенного протокола MP-BGP переносить в маршрутных объявлениях адреса разных типов, в том числе IPv6, IPX, а также VPN-IPv4. Адреса VPN-IPv4 используются только для маршрутов, которыми маршрутизаторы PE обмениваются по протоколу BGP. Прежде чем передать своему напарнику некоторый маршрут, входной маршрутизатор PE добавляет к его адресу назначения IPv4 префикс RD для данной сети VPN, тем самым преобразуя его в маршрут VPN-IPv4.

Как уже отмечалось, различители маршрута должны гарантированно уникально идентифицировать VPN, чтобы избежать дублирования адресов. Упростить выбор RD, не создавая

для этих целей дополнительных централизованных процедур (например, распределения RD органами Интернета подобно распределению адресов IPv4), предлагается за счет использования в качестве основы для RD заведомо уникальных чисел — либо номеров автономных систем, либо публичных адресов интерфейсов PE с магистральной сетью провайдера (сети провайдера всегда необходимы публичные адреса для взаимодействия с сетями других провайдеров).

На рис. 22.9 показано, как входной маршрутизатор PE1 добавляет различитель маршрутов 123.45.67.89:1 (123.45.67.89 — это глобальный адрес интерфейса маршрутизатора PE, а 1 — назначенный администратором номер) ко всем адресам с префиксом 10.1/16, которые он получает от маршрутизатора CE сайта 1 в VPN A, и пересылает эти маршруты на два других выходных маршрутизатора PE. Аналогично, маршрутизатор PE1 добавляет различитель маршрутов 123.45.67.89:2 к адресам с префиксом 10.1/16 в маршрутах, которые он получает от маршрутизатора CE сайта 1 в VPN B, и передает сформированные маршруты на другие два маршрутизатора PE. Только благодаря этим добавлениям протокол BGP, работающий на удаленных маршрутизаторах PE, способен различать маршруты с совпадающими адресами IPv4, относящиеся к разным сетям VPN.

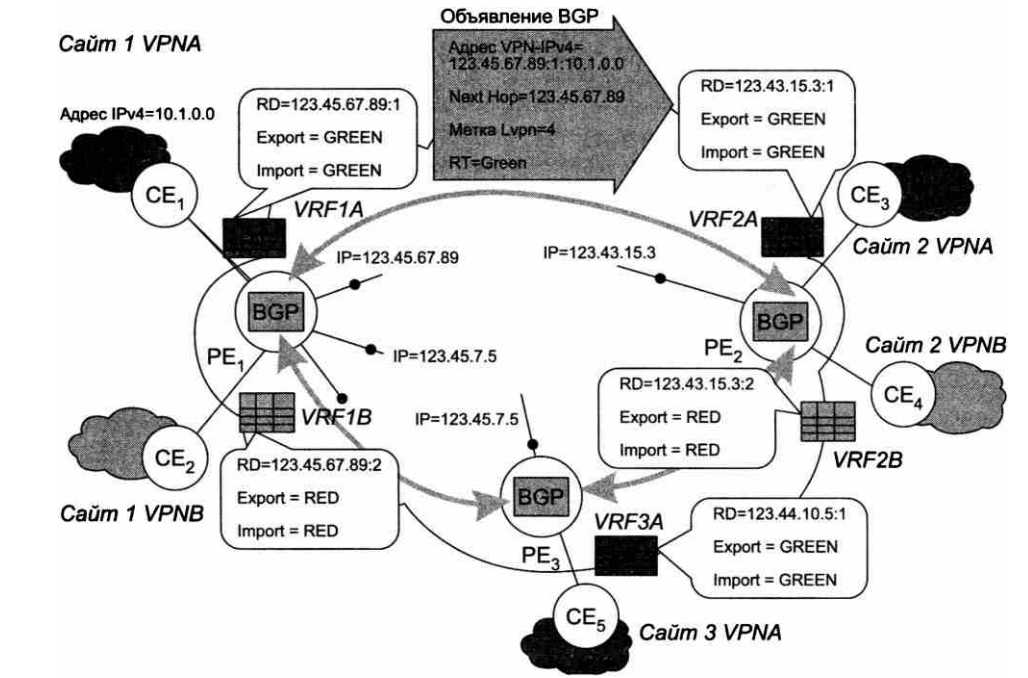


Рис. 22.9. Маршрутные объявления MB-BGP

Когда выходной маршрутизатор PE получает маршрут к сети VPN-IPv4, он делает обратное преобразование, отбрасывая префикс RD, и только потом помещает маршрут в таблицу VRF и объявляет его связанному с ним маршрутизатору заказчика CE из данной сети VPN. Таким образом, все маршруты в таблицах VRF содержат адреса в формате IPv4.

Конфигурирование топологии VPN

MPLS VPN третьего уровня позволяют создавать различные топологии связей между сайтами одной и той же сети VPN. Этим свойством сети VPN данного типа отличаются от сетей MPLS VPN второго уровня, в которых сайты одной и той же сети VPN всегда достижимы друг для друга. Например, в MPLS VPN третьего уровня можно создать звездообразную топологию, в которой периферийные сайты могут взаимодействовать с центральным сайтом, а между собой нет, — эту топологию сервис MPLS VPN второго уровня обеспечить не может.

Такая гибкая форма создания топологии VPN достигается за счет атрибутов экспорта-импорта маршрутов в объявлениях MP-BGP. Атрибут **route-target (RT)** идентифицирует входящий в данную сеть VPN набор сайтов (VRF), которым маршрутизатор PE должен посылать маршруты.

Значение атрибута route-target в объявлении о маршруте определяется политикой экспорта маршрутных объявлений, которая была задана при конфигурировании таблицы VRF, содержащей данный маршрут. Если же маршрут не входит в число экспортируемых, то он не передается другим маршрутизаторам PE, а используется локально. Такое возможно в случае, когда два маршрутизатора CE в одной и той же сети VPN непосредственно подключены к одному и тому же маршрутизатору PE. Формат атрибута route-target аналогичен формату различителя маршрутов (RD), что обеспечивает его уникальность в пределах всех сетей VPN.

При получении объявлений MP-BGP вступает в действие политика импорта маршрутов; как и политика экспорта, она задается при конфигурировании каждой таблицы VRF.

Задание одного и того же значения для политики экспорта и импорта для всех таблиц VRF определенной сети VPN приводит к полносвязной топологии — каждый сайт пересылает пакеты непосредственно тому сайту, в котором находится сеть назначения.

Именно этот случай для VPN A и VPN B показан на рис. 22.9, так как таблицы VRF сайтов этих сетей VPN сконфигурированы с одинаковыми значениями политики экспорта и импорта: значением GREEN для VPN A и значением RED для VPN B.

Пример конфигурирования звездообразной топологии представлен на рис. 22.10.

Для достижения этого эффекта достаточно определить для VRF центрального сайта политику импорта как import = spoke, экспорта — как export = hub, а для VFR периферийных сайтов поступить наоборот, задав import = hub и export = spoke. В результате таблицы VRF периферийных сайтов не смогут принимать маршрутные объявления друг от друга, поскольку они передаются по сети протоколом MP-BGP с атрибутом routetarget = spoke, между тем как их политика импорта разрешает получать объявления с атрибутом route-target = hub. Зато объявления таблиц VRF периферийных сайтов принимает таблица VRF центрального сайта, для которого как раз и определена политика импорта spoke. Этот сайт обобщает все объявления периферийных сайтов и отсылает их обратно, но уже с атрибутом route-target = hub, что совпадает с политикой импорта периферийного сайта. Таким образом, в VRF каждого периферийного сайта появляются записи о сетях в других периферийных сайтах с адресом связанного с центральным сайтом интерфейса PE в качестве следующего транзитного узла — поскольку объявление пришло от него. Поэтому пакеты между периферийными сайтами будут проходить транзитом через пограничный маршрутизатор PE3, подключенный к центральному сайту.

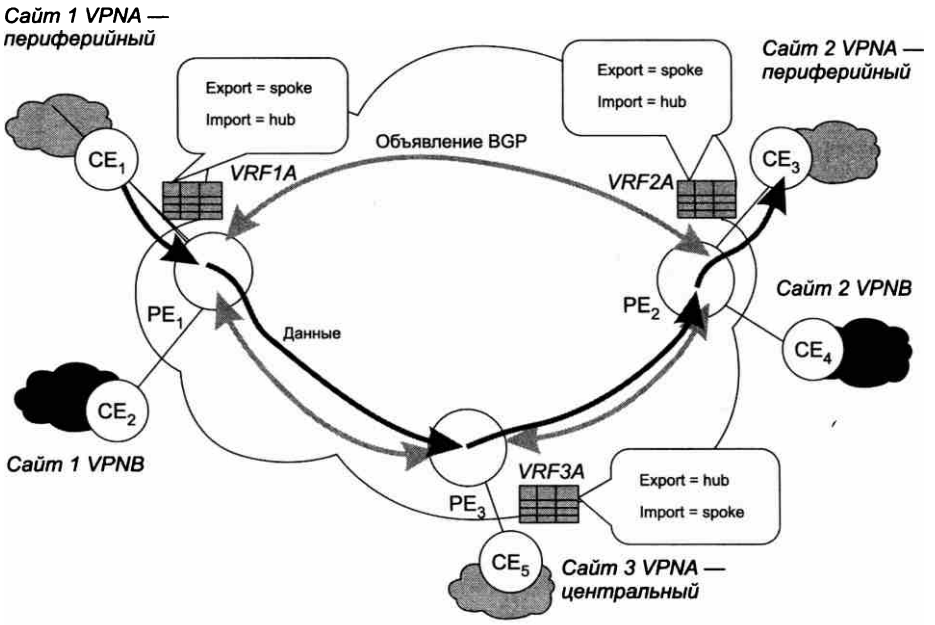


Рис. 22.10. Конфигурирование звездообразной топологии между сайтами VPNA

Выводы

Сервис виртуальных частных сетей может быть реализован различными способами и с различной степенью приближения к сервису частных сетей на выделенных каналах, который он эмулирует.

Сеть VPN может быть реализована как самим предприятием, так и поставщиком услуг. Она может строиться на базе оборудования, установленного на территории и потребителя, и поставщика услуг.

Существует несколько вариантов организации услуги VPN второго уровня:

- Ethernet поверх MPLS (EoMPLS);
- Ethernet поверх Ethernet;
- Ethernet поверх транспорта первичных сетей.

Основные потребительские свойства глобальной услуги Ethernet стандартизованы форумом MEF.

В технологии EoMPLS применяется двухуровневая иерархия соединений: на нижнем уровне работают туннели MPLS, а на верхнем — псевдоканалы, переносящие пользовательский трафик.

С помощью технологии EoMPLS провайдер может оказывать услуги двух типов: VPWS (соединения «точка-точка») и VPLS (соединения «каждый с каждым»).

Технология IP/MPLS используется при предоставлении услуги MPLS VPN третьего уровня. MPLS VPN третьего уровня поддерживают все типы топологий соединений сайтов пользователей: двухточечную, полносвязную, звездообразную.

Контрольные вопросы

1. В чем заключаются преимущества услуг виртуальных частных сетей по сравнению с услугами выделенных каналов с точки зрения поставщика этих услуг? Варианты ответов:
   а) их легче конфигурировать;
   б) можно обслужить большее число клиентов, имея ту же инфраструктуру физических каналов связи;
   в) легче контролировать соглашения SLA.
2. В чем заключаются недостатки услуг виртуальных частных сетей по сравнению с услугами выделенных каналов с точки зрения клиентов? Варианты ответов:
   а) возможны задержки и потери пакетов;
   б) не всегда есть гарантии пропускной способности соединений;
   в) высокая стоимость услуг.
3. Какие свойства частной сети имитирует услуга виртуальных частных сетей, предоставляемая провайдером? Варианты ответов:
   а) независимость адресных пространств;
   б) высокое качество обслуживания;
   в) защищенность передаваемых данных;
   г) независимость администрирования.
4. Псевдоканал MPLS — это:
   а) путь LSP второго уровня иерархии;
   б) эмулятор некоторого телекоммуникационного сервиса;
   в) путь LSP первого уровня иерархии.
5. Чем отличаются услуги L2VPN и L3VPN? Варианты ответов:
   а) при оказании услуг L2VPN в сети провайдера связи используется технология второго уровня, а при оказании услуг L3VPN — третьего;
   б) при оказании услуг L2VPN провайдер соединяет сайты клиента на основе адресной информации второго уровня, а при оказании услуг L3VPN — третьего.

Часть VI

Сетевые информационные службы

- Глава 23. Информационные службы IP-сетей
- Глава 24. Сетевая файловая служба
- Глава 25. Служба управления сетью

С точки зрения пользователей, компьютерные сети представляют собой набор служб, которые предоставляют им разнообразные услуги, таких как электронная почта, WWW, интернет-телефония, удаленный доступ к файлам, справочная служба, облачные вычисления и многие другие.

Заметим, что термин «service» в технической литературе переводится и как «сервис», и как «услуга», и как «служба». Хотя указанные термины иногда используются как синонимы, следует иметь в виду, что в некоторых случаях различие в значениях этих терминов носит принципиальный характер. Мы понимаем под «службой» сетевой программный компонент, который реализует предоставление некоторого набора услуг, а под «сервисом» — собственно тот набор услуг, которой предоставляется службой. То есть термины «сервис» и «услуга» используются далее как синонимы.

Все перечисленные службы предоставляют *прикладные*, они же *информационные*, сервисы и реализуются программным обеспечением, работающим на конечных узлах сети — компьютерах. Очевидно, что прикладные сервисы не могут предоставляться без обеспечивающих их транспортных сервисов, но эти функции сети скрыты от конечных пользователей.

В то же время характеристики *транспортных* сервисов, доставляющих информацию от одного конечного узла сети другому, существенно влияют на качество прикладных сервисов: например, доставка пакетов интернет-телефонии с задержками более 100 мс сделает голос собеседника неузнаваемым, а предоставление видеосервису пропускной способности менее 5 Мбит/с приведет к многочисленным остановкам изображения высокого разрешения. Именно требования прикладных сетевых сервисов являются движущей силой всех изменений и усовершенствований в области транспортных технологий. Рост популярности мультимедийных сервисов реального времени (потоковое видео, аудио) привел к появлению новых протоколов транспортного уровня, переносящих временны́е отметки в отправленных пакетах, позволяющие приемной стороне контролировать темп поступления данных. Другим следствием растущей популярности мультимедийных сервисов стал взрывной рост объемов трафика, переносимого Интернетом, что делает насущным дальнейший рост производительности транспортных сервисов этой глобальной сети.

В этой части книги вы познакомитесь с организацией наиболее популярных прикладных сетевых служб, предоставляющих свои услуги конечным пользователям: почтой, веб-службой, IP-телефонией, файловой и справочной службами, сетевым управлением на основе протокола SNMP и облачными вычислениями.

Некоторые службы, а именно службы поддержки транспортных средств сети DNS и DHCP, выполняющие адресацию сетевых узлов, рассмотрены в предыдущих главах. Это было сделано из-за их тесной связи с работой стека протоколов TCP/IP.

Существует еще одна группа прикладных сетевых служб, которая обеспечивает безопасность сети, выполняя аутентификацию и авторизацию пользователей, шифрование трафика и ряд других важных операций. Эта группа служб рассматривается в последней части книги, целиком посвященной средствам обеспечения безопасности. Там вы также найдете описание проблем безопасности, специфических для каждой из прикладных служб, ориентированных на конечного пользователя, то есть проблем безопасности веб-сервисов, почтовой службы и др.

ГЛАВА 23 Информационные службы IP-сетей

Общие принципы организации сетевых служб

Службы принято делить на несколько групп по типам адресатов предоставляемых ими услуг:

- Службы, ориентированные на конечных пользователей и их приложения, такие как служба печати, файловый сервис, почта, веб-сервис, справочная служба, IP-телефония, служба облачных вычислений.
- Службы, обеспечивающие безопасность сети, — к ним относятся сетевая аутентификация, авторизация и контроль доступа. Услуги этих служб требуются как конечным пользователям, например при интерактивном входе в сеть, так и другим службам, которым необходимо защитить свою информацию и аутентифицировать своих пользователей, — примером может быть служба баз данных, аутентифицирующая своих пользователей с помощью службы сетевой аутентификации.
- Службы, ориентированные на сетевых администраторов, решающих задачи конфигурирования и управления сетевыми устройствами; в эту категорию входят службы управления сетью на основе протоколов telnet и SNMP, служба мониторинга и аудита.
- Службы, помогающие компьютерам и сетевым устройствам предоставлять свои транспортные услуги, такие как служба отображения символьных имен узлов на IP-адреса (DNS) и служба динамического назначения адресов (DHCP).
- Службы поддержки распределенных вычислений: например, служба репликации, служба вызова удаленных процедур (RPC), являющиеся вспомогательными по отношению к другим службам.

Клиентами сетевых служб могут быть другие сетевые службы: например, в число клиентов справочной службы входят служба аутентификации и почтовая служба.

В то же время служба помимо основных услуг, дающих имя этому типу службы, может предоставлять и вспомогательные услуги. Например, веб-служба может выполнять аутентификацию самостоятельно, не обращаясь к централизованной службе сетевой аутентификации.

Большая часть прикладных сетевых служб оформляются как приложения, то есть в виде исполняемых модулей стандартного для ОС, в среде которой они выполняются, формата. Поскольку такой же формат имеют и многие модули ОС, то часто бывает сложно провести четкую грань между операционной системой и сетевыми службами. Решение о том, должна ли какая-то служба стать частью ОС или нет, принимает производитель данной ОС.

В некоторых случаях для повышения производительности сервиса служба или ее определенные компоненты включаются в ядро. Примером являются клиентская и серверная части файловой службы, которые часто встраивают в ядро с тем, чтобы они могли получать быстрый прямой доступ ко всем модулям ОС без затрат времени на переключение режима из пользовательского в привилегированный.

Сетевые службы чаще всего представляют собой двухзвенные[1] *распределенные приложения*, одно из звеньев является клиентом, другое — сервером. Клиентская и серверная части в общем случае выполняются на разных компьютерах. Как правило, один сервер обслуживает большое число клиентов.

Принципиальной разницей между клиентом и сервером является то, что инициатором выполнения сетевой службой некой работы всегда выступает клиент, а сервер всегда находится в режиме пассивного ожидания запросов. Например, почтовый сервер осуществляет доставку почты на компьютер пользователя только при поступлении запроса от почтового клиента.

В отличие от локальных приложений, которые, работая на одном компьютере, могут обмениваться данными через его оперативную память, части распределенного приложения, выполняемые на разных компьютерах, такой возможности не имеют. Взаимодействие клиента и сервера может выполняться только путем передачи *сообщений* через сеть в соответствии с выбранным *протоколом*.

Основными вопросами разработки распределенных приложений являются, во-первых, распределение функций между его звеньями (клиентом и сервером), а во-вторых, определение протокола взаимодействия этих звеньев.

Распределение функций между клиентом и сервером сетевой службы может выполняться различными способами. Например, клиент может быть наделен только функциями поддержки интерфейса с пользователем сервиса и поддержанием протокола взаимодействия, а вся логика работы службы возложена на серверную часть. Возможна и другая ситуация, когда клиент несет значительную нагрузку на поддержание работы сетевой службы. Например, при реализации почтовой службы на диске клиента может храниться локальная копия базы данных, содержащей его обширную переписку. В этом случае клиент делает основную работу при формировании сообщений в различных форматах, в том числе и сложном мультимедийном, поддерживает ведение адресной книги и выполняет еще много различных вспомогательных функций.

Протоколы обмена сообщениями, лежащие в основе сетевых служб, относятся к прикладному уровню. Службы, имеющие одно и то же назначение, могут использовать разные протоколы. К примеру, существуют сетевые файловые системы, построенные на основе принципиально отличающихся протоколов: FTP, SMB, NFS. Аналогично, имеются два типа почтовой службы, в одной из них клиент и сервер взаимодействуют по протоколу SMTP, а в другой — X.400. Верно и обратное утверждение: службы, разработанные для предоставления разных сервисов, могут использовать один и тот же протокол взаимодействия клиентской и серверной частей. Например, протокол HTTP, разработанный для веб-службы, стал использоваться во многих службах и сетевых приложениях, например в службе управления сетью, в почтовой службе и многих других.

---

[1] Распределенные приложения вообще и сетевые службы в частности могут иметь и многозвенную структуру.

Пользовательский интерфейс, который в наше время обычно является графическим (Graphical User Interface, GUI), — важная часть клиента сетевой службы. От качества этого интерфейса зависит удобство работы пользователя с данной реализацией службы (степень дружественности), он также отражает функциональное богатство службы. Нужно заметить, что различные реализации службы одного и того же назначения, поддерживающие один и тот же прикладной протокол, могут значительно отличаться друг от друга функциональностью и дружественностью пользовательского интерфейса. Это хорошо видно на примере браузеров веб-службы — все они работают с одними и теми же веб-серверами и реализуют один и тот же протокол HTTP, но функциональность и дружественность браузера Chrome отличается от функциональности и дружественности браузера Internet Explorer (хотя жесткая конкуренция между производителями браузеров заставляет их постоянно перенимать лучшие свойства продуктов конкурентов).

Веб-служба

Изобретение в 1989 году Тимом Бернерсом-Ли и Робертом Кайо **Всемирной паутины** (World Wide Web, **WWW**) стоит в одном ряду с изобретениями телефона, радио и телевидения. Благодаря этому изобретению Интернет стал таким, каким мы его знаем сегодня.

Используя Всемирную паутину, люди получили возможность доступа к нужной им информации в любое удобное для них время. Теперь проще найти интересующую вас статью в Интернете, чем в стопке журналов, хранящихся рядом в шкафу. Очень быстро исчезают многие традиционные приемы рациональной организации работы с информацией, заключающиеся, например, в хранении полезной информации в записных книжках, раскладывании вырезок из журналов и газет в картонные папки с веревочками, упорядочивании документов в каталогах путем наклеивания на них маркеров с условными кодами, помогающими быстро отыскать нужный документ, и т. д. Этим приемам приходят на смену новые безбумажные технологии Интернета, среди которых важнейшей является сетевая **служба WWW** (или **веб-служба**).

Заметим, что веб-служба не только предоставляет любому человеку возможность быстрого поиска нужных данных и доступа к ним, но и позволяет ему выносить на многомиллионную аудиторию пользователей Интернета собственную информацию — мнения, художественные и публицистические произведения, результаты научной работы, объявления и т. д. Причем он может это делать без особых организационных забот и практически бесплатно. Мы не будем долго останавливаться на описании всех возможностей этой службы, учитывая, что для большинства из нас регулярный просмотр веб-сайтов стал не просто обыденностью, а необходимым элементом жизненного уклада.

Веб- и HTML-страницы

Миллионы компьютеров, связанных через Интернет, хранят невообразимо огромные объемы информации, представленной в виде веб-страниц.

Веб-страница, или **веб-документ**, как правило, состоит из основного HTML-файла и некоторого количества ссылок на другие объекты разного типа: JPEG- и GIF-изображения, другие HTML-файлы, аудио- и видеофайлы.

HTML-файлом, **HTML-страницей**, или **гипертекстовой страницей**, называют файл, который содержит текст, написанный на **языке HTML** (HyperText Markup Language — язык разметки гипертекста).

История появления языка HTML связана с попытками программистов разработать средство, которое бы позволяло им программным путем создавать красиво сверстанные страницы для просмотра на экране. Другими словами, красивая картинка появляется на дисплее только в результате ее интерпретации специальной программой, а в исходном виде она представляет собой однообразный текст с множеством служебных пометок. Вместо применения различных приемов форматирования, таких как выделение заголовков крупным шрифтом, а важных выводов — курсивным или полужирным начертанием, создатель документа на языках этого типа просто вставляет в текст соответствующие указания о том, что данная часть текста должна быть выведена на экран в том или ином виде. Служебные пометки такого рода в исходном тексте выглядят, например, как `<b> </b>` (начать и закончить вывод текста полужирным начертанием) и называются **тегами**. Язык HTML не является первым языком разметки текста, его предшественники существовали задолго до появления веб-службы: например, в первых версиях ОС Unix существовал язык troff (с помощью этого языка отформатированы страницы электронной документации Unix, известные как man-страницы).

В язык HTML включены разные типы тегов, команд и параметров, в том числе для вставки в текст изображений (тег `<img src='...'`). Чтобы HTML-страница выглядела так, как задумал программист, она должна быть выведена на экран специальной программой, способной интерпретировать язык HTML. Такой программой является уже упоминавшийся веб-браузер.[1]

Существует особый тип тега, который имеет вид `<a href="..." ...</a>` и называется **гиперссылкой**. Гиперссылка содержит информацию о веб-странице или объекте, который может находиться как на том же компьютере, так и на других компьютерах Интернета. Отличие гиперссылки от других тегов состоит в том, что элемент, описываемый ею, не появляется автоматически на экране, вместо этого на месте тега (гиперссылки) на экран выводится некоторое условное изображение или особым образом выделенный текст — имя гиперссылки. Чтобы получить доступ к объекту, на который указывает эта гиперссылка, пользователь должен «щелкнуть» на ней, дав тем самым команду браузеру найти и вывести на экран требуемую страницу или объект. После того как новая веб-страница будет загружена, пользователь сможет перейти по следующей гиперссылке, — такой «веб-серфинг» может продолжаться теоретически сколь угодно долго. Все это время веб-браузер будет находить указанные в гиперссылках страницы, интерпретировать все размещенные на них указания и выводить информацию на экран в том виде, в каком ее спроектировали разработчики этих страниц.

URL-адрес

Браузер находит веб-страницы и отдельные объекты по адресам специального формата, называемым **URL** (Uniform Resource Locator — унифицированный указатель ресурса). URL-адрес может выглядеть, например, так: http://www.olifer.co.uk/books/books.htm.

---

[1] См. раздел «Сетевое программное обеспечение» в главе 2.

В URL-адресе можно выделить три части:

- *Тип протокола доступа*. Помимо HTTP, здесь могут быть указаны и другие протоколы, такие как FTP, telnet, также позволяющие осуществлять удаленный доступ к файлам или компьютерам[1]. Тем не менее основным протоколом доступа к веб-страницам является HTTP (как в нашем примере), и мы поговорим о нем немного позже.
- *DNS-имя сервера*. Это имя сервера, на котором хранится нужная страница. В нашем случае — это имя сайта www.olifer.co.uk.
- *Путь к объекту*. Обычно это составное имя файла (объекта) относительно главного каталога веб-сервера, предлагаемого по умолчанию. В нашем случае главным каталогом является /books/books.htm. По расширению файла мы можем сделать вывод о том, что это HTML-файл.

Веб-клиент и веб-сервер

Как мы уже отмечали, сетевая веб-служба представляет собой распределенную программу, построенную в архитектуре клиент-сервер. Клиент и сервер веб-службы взаимодействуют друг с другом по протоколу HTTP.

Клиентская часть веб-службы, или веб-клиент, называемый также браузером, представляет собой приложение, которое устанавливается на компьютере конечного пользователя и предназначено для просмотра веб-страниц.

Одной из важных функций браузера является *поддержание графического пользовательского интерфейса*. Через этот интерфейс пользователь получает доступ к широкому набору услуг, главная из которых, конечно, — «веб-серфинг», включающий поиск и просмотр страниц, навигацию между уже просмотренными страницами, переход по закладкам и хранение истории посещений. Помимо средств просмотра и навигации, веб-браузер предоставляет пользователю возможность *манипулирования страницами*: сохранение их в файле на диске своего компьютера, вывод на печать, передача по электронной почте, контекстный поиск в пределах страницы, изменение кодировки и формата текста, а также множество других функций, связанных с представлением информации на экране и настройкой самого браузера.

К числу наиболее популярных сейчас браузеров можно отнести Microsoft Internet Explorer, Mozilla Firefox компании Mozilla, Google Chrome и Apple Safari. Веб-браузер — это не единственный вид клиента, который может обращаться к веб-серверу. Эту роль могут исполнять любые программы и устройства, поддерживающие протокол HTTP.

Значительную часть своих функций браузер выполняет в тесной кооперации с веб-сервером. Как уже отмечалось, клиент и сервер веб-службы связываются через сеть по протоколу HTTP. Это означает, что в клиентской части веб-службы присутствует клиентская часть HTTP, а в серверной — серверная часть HTTP.

---

[1] URL-адреса с самого начала предназначались не только для веб-служб, но и для других сервисов доступа к информации через Интернет.

Веб-сервер — это программа, хранящая объекты локально в каталогах компьютера, на котором она запущена, и обеспечивающая доступ к этим объектам по URL-адресам. Наиболее популярными веб-серверами сейчас являются Apache и Microsoft Internet Information Server.

Как и любой другой сервер, веб-сервер должен быть постоянно в активном состоянии, прослушивая *TCP-порт 80*, который является назначенным портом протокола HTTP. Как только сервер получает запрос от клиента, он устанавливает TCP-соединение и получает от клиента имя объекта, например в виде /books/books.htm, после чего находит в своем каталоге этот файл, а также другие связанные с ним объекты и отсылает по TCP-соединению клиенту. Получив объекты от сервера, веб-браузер отображает их на экране (рис. 23.1). После отправки всех объектов страницы клиенту сервер разрывает с ним TCP-соединение. В дополнительные функции сервера входят также аутентификация клиента и проверка прав доступа данного клиента к данной странице.

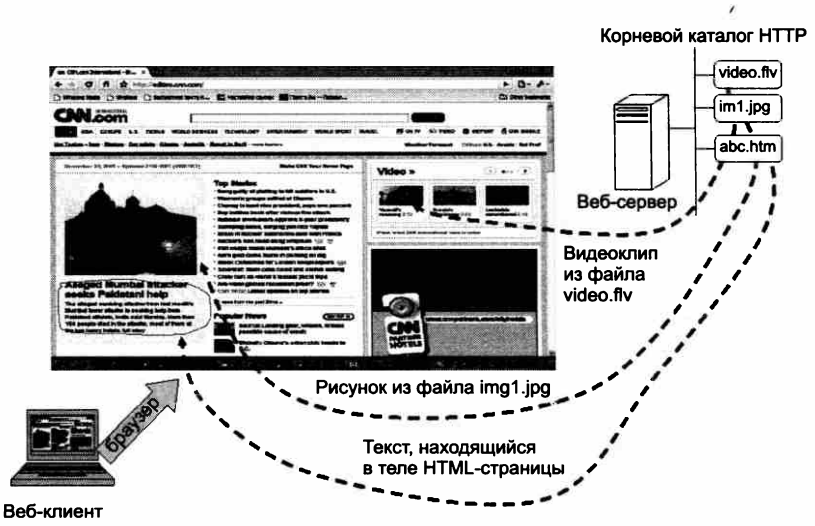


Рис. 23.1. Вывод веб-страницы на экран

Веб-сервер в отношении сеанса с веб-браузером является *сервером без сохранения состояния* (stateless). Это означает, что на сервере не хранится информация, касающаяся состояния сеанса: какие страницы пользователь уже посетил и какие данные ему были переданы. Такой режим общения с клиентом упрощает организацию сервера, которому необходимо отвечать на большой поток запросов различных пользователей, так что запоминание состояния сеансов пользователей существенно увеличило бы нагрузку на веб-сервер. Вместо этого веб-сервер рассматривает каждый запрос изолированно, отвечая на него и забывая про данного пользователя сразу после ответа. Кроме упрощения организации сервера, такой режим работы является более устойчивым, так как он не требует восстановления сеанса, если по той или иной причине он потерпел крах, оставляя пользователю заботы по решению этой проблемы. Недостатком данного режима является замедление работы клиента и увеличение трафика в сети из-за частого выполнения процедуры установления TCP-соединений.

Для повышения производительности некоторые веб-серверы прибегают к кэшированию наиболее часто используемых в последнее время страниц в своей памяти. Когда приходит запрос на какую-либо страницу, сервер, прежде чем считывать ее с диска, проверяет, не находится ли она в буферах более «быстрой» оперативной памяти. Кэширование страниц осуществляется и на стороне клиента, а также на промежуточных серверах (прокси-серверах). Кроме того, эффективность обмена данными с клиентом иногда повышают путем компрессии (сжатия) передаваемых страниц. Объем передаваемой информации уменьшают также за счет того, что клиенту передается не весь документ, а только та часть, которая была изменена. Все эти приемы повышения производительности веб-службы реализуются средствами протокола HTTP.

Протокол HTTP

HTTP (HyperText Transfer Protocol — протокол передачи гипертекста) — это протокол прикладного уровня, во многом аналогичный протоколам FTP и SMTP. В настоящее время используются две версии протокола: HTTP 1.0 и HTTP 1.1.

Обмен сообщениями идет по обычной схеме «запрос-ответ». Клиент и сервер обмениваются *текстовыми* сообщениями стандартного формата, то есть каждое сообщение представляет собой несколько строк обычного текста в кодировке ASCII.

Для транспортировки HTTP-сообщений служит протокол TCP. При этом TCP-соединения могут использоваться двумя разными способами:

- *долговременное соединение* — передача в одном TCP-соединении нескольких объектов, причем время существования соединения определяется при конфигурировании веб-службы;
- *кратковременное соединение* — передача в рамках одного TCP-соединения только одного объекта.

Долговременное соединение, в свою очередь, может быть использовано двумя способами:

- *последовательная передача запросов с простоями* — новый запрос посылается только после получения ответа;
- *конвейерная передача* — это более эффективный способ, в котором следующий запрос посылается до прибытия ответа на один или несколько предыдущих запросов (напоминает метод скользящего окна). Обычно по умолчанию степень параллелизма устанавливается на уровне 5–10, но у пользователя имеется возможность изменять этот параметр при конфигурировании клиента.

В версии HTTP 1.0 поддерживается только режим кратковременных соединений, когда после передачи одного запроса и получения ответа TCP-соединение закрывается. Такой режим полностью соответствует концепции сервера без сохранения состояния, а это, как уже отмечалось, приводит к замедлению работы браузера и увеличению трафика из-за частого выполнения процедуры трехэтапного установления TCP-соединения.

В версии HTTP 1.1 по умолчанию применяются постоянные соединения и конвейерный режим. Соединение разрывается по инициативе либо браузера, либо сервера за счет отправки специального токена разрыва соединения в HTTP-пакете. Веб-сервер обычно использует таймер неактивности пользователя для того, чтобы разорвать соединение по тайм-ауту и не тратить ресурсы памяти на неактивные соединения.

Формат HTTP-сообщений

В протоколе HTTP все сообщения состоят из текстовых строк. HTTP-сообщения бывают двух типов: запросы и ответы. Запросы и ответы имеют единую обобщенную структуру, состоящую из трех частей: обязательной стартовой строки, а также необязательных заголовков и тела сообщения. В табл. 23.1 приведены форматы и примеры стартовых строк и заголовков для запросов и ответов.

Таблица 23.1. Форматы стартовых строк и заголовков

Обобщенная структура сообщения	**HTTP-запрос**	**HTTP-ответ**
Стартовая строка (всегда должна быть первой строкой сообщения; обязательный элемент)	Формат запроса Метод/ URL HTTP/1.x. Пример: GET /books/ books.htm HTTP/1.1	Формат ответа: HTTP/1.x КодСостояния Фраза. Пример: HTTP/1.1 200 OK
Заголовки (следуют в произвольном порядке; могут отсутствовать)	Заголовок о DNS-имени компьютера, на котором расположен веб-сервер. Пример: Host: www.olifer.co.uk	Заголовок о времени отправления данного ответа. Пример: Date: 1 Jan 2009 14:00:30
	Заголовок об используемом браузере. Пример: User-agent: Mozilla/5.0	Заголовок об используемом веб-сервере. Пример: Server: Apache/1.3.0 (Unix)
	Заголовок о предпочтительном языке. Пример: Accept-language: ru	Заголовок о количестве байтов в теле сообщения. Пример: Content-Length: 1234
	Заголовок о режиме соединения. Пример: Connection: close	Заголовок о режиме соединения. Пример: Connection: close
Пустая строка		
Тело сообщения (может отсутствовать)	Здесь могут быть расположены ключевые слова для поисковой машины или страницы для передачи на сервер	Здесь может быть расположен текст запрашиваемой страницы

Как видно из таблицы, запросы и ответы имеют разные форматы стартовой строки. Каждая из них состоит из трех элементов, включающих поле *версии протокола HTTP*. И в запросе, и в ответе примера указана версия HTTP 1.1. Стартовая строка запроса включает в себя поле *метода* — это название операции, которая должна быть выполнена. Чаще всего в запросах используется метод GET, то есть запрос объекта. Именно он включен в наш пример запроса.

Еще одним элементом стартовой строки является URL-адрес запрашиваемого объекта — здесь это имя файла /books/books.htm.

Помимо метода GET в запросах протокол предусматривает и другие методы, такие как HEAD, POST, PUT, DELETE и некоторые другие.

Метод HEAD аналогичен методу GET, но запрашиваются только метаданные заголовка HTML-страницы.

Метод POST используется клиентом для отправки данных на сервер: сообщений электронной почты, ключевых слов в запросе поиска, веб-формы.

Метод PUT используется клиентом для размещения объекта на сервере, на который указывает URL-адрес.

Метод DELETE указывает серверу на то, что некоторый объект на сервере, определяемый URL-адресом, необходимо удалить.

Методы GET и HEAD считаются безопасными[1] для сервера, так как они только передают информацию клиенту, а методы POST, PUT и DELETE — опасными, поскольку передают информацию на сервер. Наибольшую угрозу представляют два последних метода, так как они непосредственно указывают на объект на сервере. Используя эти методы, злоумышленник может атаковать сервер, заменяя или удаляя некоторые его объекты.

В стартовой строке ответа, помимо уже упоминавшегося указания на версию протокола HTTP, имеется поле *кода состояния* и поле *фразы* для короткого текстового сообщения, поясняющего данный код пользователю.

В настоящее время стандарты определяют пять классов кодов состояния:

- 1*xx* — информация о процессе передачи;
- 2*xx* — информация об успешном принятии и обработке запроса клиента (в таблице в примере стартовой строки ответа приведен код и соответствующая фраза `200 OK`, сообщающий клиенту, что его запрос успешно обработан);
- 3*xx* — информация о том, что для успешного выполнения операции нужно произвести следующий запрос по другому URL-адресу, указанному в дополнительном заголовке `Location`;
- 4*xx* — информация об ошибках на стороны клиента (читатель наверняка не раз сталкивался с ситуацией, когда при указании адреса несуществующей страницы браузер выводил на экран сообщение `404 Not Found`);
- 5*xx* — информация о неуспешном выполнении операции по вине сервера (например, сообщение `505 http Version Not Supported` говорит о том, что сервер не поддерживает версию HTTP, предложенную клиентом).

Среди кодов состояния имеется код 401, сопровождаемый сообщением `authorization required`. Если клиент получает такое сообщение в ответ на попытку доступа к странице или объекту, это означает, что доступ к данному ресурсу ограничен и требует авторизации пользователя. Помимо поясняющей фразы сервер помещает в свой ответ дополнительный заголовок `www-Authenticate:<...>`, который сообщает клиенту, какую информацию он должен направить серверу для того, чтобы процедура авторизации могла быть выполнена. Обычно это имя и пароль. Веб-клиент с момента получения такого ответа сервера начинает добавлять во все свои запросы к ресурсам данного сервера дополнительный заголовок `Authorization: <`*имя*, *пароль*`>`, который содержит информацию, необходимую для авторизации доступа.

---

[1] Вопросы безопасности веб-службы подробно обсуждаются в главе 30.

Динамические веб-страницы

До сих пор мы подразумевали, что содержание страницы не изменяется в зависимости от действий пользователя. Когда пользователь щелкает на гиперссылке, то он переходит на *новую* страницу, а если выполняет команду возвращения обратно, то на экране снова появляется предыдущая страница в *неизменном* виде. Такие страницы называются **статическими**.

Однако в некоторых случаях желательно, чтобы содержание страницы изменялось в зависимости от действий пользователя: например, при наведении указателя мыши на определенную область страницы там появлялся рисунок вместо текста или значка. Динамическое воспроизведение состояния базы данных также является типичным примером ситуации, когда статическая страница не может решить задачу. Например, многие интернет-магазины поддерживают базу данных продаваемых товаров, и вывод количества оставшихся в наличии товаров требует динамического обновления соответствующего поля веб-страницы.

Веб-страницы, которые могут генерировать выводимое на экран содержание, меняющееся в зависимости от некоторых внешних условий, называются **динамическими**.

Динамика страницы достигается путем ее программирования, обычно для этого используются программные языки сценариев, такие как Perl, PHP и JavaScript.

Различают два класса программ, предназначенных для создания динамического содержания веб-страниц:

- программы, работающие на стороне клиента (то есть на том компьютере, где запущен веб-браузер, воспроизводящий страницу на экране);
- программы, работающие на стороне сервера.

В том случае, когда программа работает на стороне клиента, код страницы передается веб-сервером веб-браузеру как обычный статический объект, а затем браузер выполняет этот код, с его помощью создает динамическое содержание страницы и выводит ее на экран.

Существуют различные механизмы создания динамических страниц на стороне клиента — это, прежде всего, надстройки браузера, Java-апплеты, JavaScript-сценарии и ActiveX-элементы.

Механизм *надстроек браузера* (add-on) позволяет расширить его функциональные возможности за счет динамического вызова из браузера дополнительных программ, установленных на клиентском компьютере. Программа-надстройка обрабатывает объекты веб-страницы определенного типа: например, программа-надстройка Adobe Acrobat NPAPI Plugin вызывается браузером Firefox для показа пользователю документа в формате PDF в окне браузера, а программа-надстройка Shockwave Flash вызывается для проигрывания видеоклипов в формате Flash или для интерактивной анимации, написанной на языке ActionScript. Нужно отметить, что термин «надстройка» считается обобщенным названием *разных* видов надстроек браузера: например, браузера Firefox, который различает несколько видов надстроек:

- *расширения* (extensions) встраиваются в браузер (то есть становятся его частью);
- *темы* изменяют внешний вид окна браузера и также встраиваются в него;
- *вставки* (plug-ins) — это программы, оформленные чаще всего в виде библиотек и вызываемые через стандартный для браузера интерфейс, такой как, например, NPAPI (Netscape Plugin Application Programming Interface). Вставки являются внешними по отношению к браузеру программами.

Java-апплеты представляют собой скомпилированные программы, которые написаны на языке Java, динамически загружаются браузером с веб-сервера и выполняются виртуальной Java-машиной (JVM) клиентского компьютера. Передача Java-апплета Java-машине выполняется вставкой Java Applet Plugin.

Динамическое содержание страницы может также создаваться с помощью JavaScript-сценариев — специальных программ (скриптов), которые пишутся на языке JavaScript, разработанном компанией Netscape. JavaScript, хотя и имеет общую часть в названии с языком Java, является самостоятельным языком со своим синтаксисом. В отличие от Java — языка компилирующего типа — JavaScript является языком интерпретирующего типа, интерпретатором которого выступает браузер.

Динамическое содержание страницы может быть также создано предложенными компанией Microsoft управляющими ActiveX-элементами, которые могут вызывать внешние объекты и встраивать результаты их работы в страницу. ActiveX-элементы являются двоичными исполняемыми файлами, которые должны иметь цифровую подпись. Ограничением ActiveX-элементов является то, что их выполнение возможно только в среде ОС Windows на процессоре Intel x86 или же при их эмуляции. На практике это означает, что страницы с ActiveX-элементами правильно воспроизводятся только браузером Internet Explorer.

При программировании содержания страницы на стороне сервера процесс выглядит немного сложнее, так как программный код страницы создает содержание (контент) на сервере, следовательно, здесь нужен дополнительный этап — передача этого содержания по протоколу HTTP на клиентскую машину браузеру. Популярными языками сценариев для серверной части являются Perl, ASP, JSP и PHP. Существует также стандартный программный интерфейс между веб-сервером и программами, генерирующими динамическое содержание, — это общий шлюзовой интерфейс (Common Gateway Interface, CGI).

Почтовая служба

Сетевая почтовая служба, или **электронная почта**, — это распределенное приложение, главной функцией которого является предоставление пользователям сети возможности обмениваться электронными сообщениями.

Как и все сетевые службы, электронная почта построена в архитектуре клиент-сервер. Почтовый клиент всегда располагается на компьютере пользователя, а почтовый сервер, как правило, работает на выделенном компьютере.

Почтовый клиент (называемый также **агентом пользователя**) — это программа, предназначенная для поддержания пользовательского интерфейса (обычно графического), а также для предоставления пользователю широкого набора услуг по подготовке электронных сообщений. В число таких услуг входит создание текста в различных форматах и кодировках, сохранение, уничтожение, переадресация, сортировка писем по разным критериям, просмотр перечня поступивших и отправленных писем, грамматическая и синтаксическая проверка текста сообщений, ведение адресных баз данных, автоответы, образование групп рассылки и прочее, и прочее. Кроме того, почтовый клиент поддерживает взаимодействие с серверной частью почтовой службы.

Почтовый сервер выполняет прием сообщений от клиентов, для чего он постоянно находится в активном состоянии. Кроме того, он выполняет буферизацию сообщений, распределение поступивших сообщений по индивидуальным буферам (почтовым ящикам) клиентов, управляет объемами памяти, выделяемой клиентам, выполняет регистрацию клиентов и регламентирует их права доступа к сообщениям, а также решает много других задач.

Электронные сообщения

Почтовая служба оперирует **электронными сообщениями** — информационными структурами определенного стандартного формата. Упрощенно электронное сообщение может быть представлено в виде двух частей, одна из которых (заголовок) содержит вспомогательную информацию для почтовой службы, другая часть (тело сообщения) — это собственно то «письмо», которое предназначается для прочтения, прослушивания или просмотра адресатом (RFC 822).

Главными элементами заголовка являются адреса отправителя и получателя в виде Polina@domen.com, где Polina — идентификатор пользователя почтовой службы, а domen.com — имя домена, к которому относится этот пользователь. Кроме этого, почтовая служба включает в заголовок дату и тему письма, делает отметки о применении шифрования, срочности доставки, необходимости подтверждения факта прочтения этого сообщения адресатом и др. Дополнительная информация заголовка может оповещать почтового клиента получателя об использовании той или иной кодировки.

При транспортировке через Интернет почтовое сообщение помещается в **конверт** (envelope), который также имеет несколько служебных полей, например поле отправителя и поля получателей (рис. 23.2). Информация конверта используется только при транспортировке почтового сообщения, а информация заголовка сообщения — почтовым клиентом получателя.

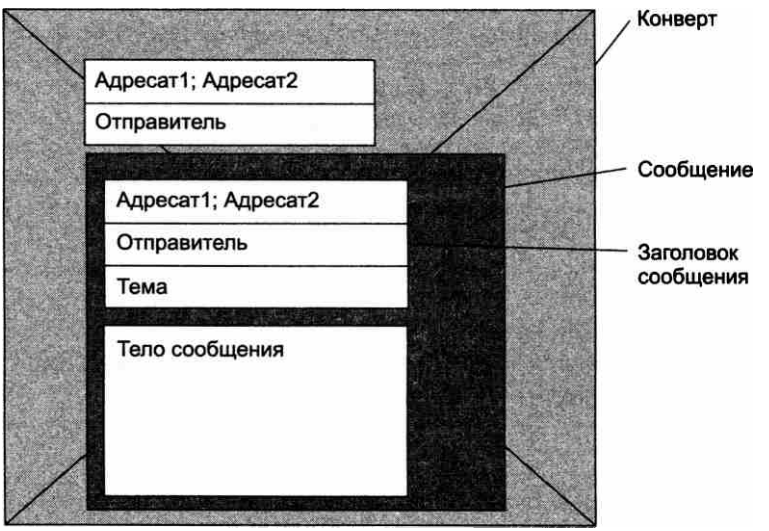


Рис. 23.2. Конверт и сообщение электронной почты Интернета

Первоначально тело сообщения представляло собой сплошной текст в кодировке ASCII, такая же кодировка использовалась для служебных полей конверта и заголовка сообщения.

Большая популярность электронной почты привела к ее интернационализации, что заставило принять несколько новых стандартов, разрешающих применять в теле сообщения не только ASCII-коды, но и такие коды, как UTF-8, позволяющие пользователям всех стран задействовать свой родной язык при написании электронного письма. Ограничение на использование ASCII осталось только для полей конверта, и это ограничение приходится преодолевать несколько искусственным способом, преобразуя исходные символы, не относящиеся к кодировке ASCII, в более длинную последовательность ASCII-кодов (например, используя популярный в системах электронной почты алгоритм base64).

Важную роль в расширении возможности электронной почты по передаче мультимедийной информации сыграл стандарт **MIME** (Multipurpose Internet Mail Extensions — многоцелевые расширения почты Интернета). Этот стандарт описывает структуру сообщения, состоящего из нескольких частей, каждая из которых имеет свои заголовок и тело. Заголовок описывает тип данных, которые содержатся в теле; это могут быть как обычные текстовые данные в формате ASCII, так и данные другого типа, например:

- текст в 8-битном формате (такая возможность стала стандартной совсем недавно, она описана в документе RFC 6152, принятом в марте 2011 года);
- текст не в формате ASCII, преобразованный в ASCII-код (например, с помощью уже упомянутого алгоритма base64);
- гипертекст (HTML);
- изображение;
- видеоклип;
- звуковой файл.

Части отделяются друг от друга последовательностью символов, называемой границей (boundary); граница не должна встречаться в теле частей сообщения.

В заголовке каждой части сообщения имеется также информация о том, каким образом почтовый клиент должен обрабатывать тело части: отображать ее немедленно при открытии сообщения (например, встраивая изображение в текст) или считать это тело *вложением* (attachment), которое пользователь будет обрабатывать сам.

Одна из спецификаций стандарта MIME (каждая спецификация MIME описывает одно или несколько расширений оригинальной спецификации RFC 822), а именно RFC 1847, относится к расширениям безопасности, поэтому этот документ называют спецификацией **S/MIME** (*Security MIME*). В S/MIME описаны два новых типа частей MIME:

- цифровая подпись (Mutipart/Signed);
- шифрованное тело (Multipart/Encrypted).

Эти два типа частей сообщения могут использоваться вместе с целью обеспечения аутентичности, целостности и конфиденциальности электронного письма.

Протокол SMTP

В качестве средств передачи сообщения почтовая служба Интернета использует стандартный, разработанный специально для почтовых систем протокол **SMTP** (*Simple Mail*

Transfer Protocol — простой протокол передачи почты). Этот протокол является одним из первых стандартизованных протоколов прикладного уровня, дата его публикации — август 1982 года.

Как и большинство других протоколов прикладного уровня, SMTP реализуется несимметричными взаимодействующими частями: SMTP-клиентом, работающим на стороне отправителя, и SMTP-сервером, работающим на стороне получателя. SMTP-сервер должен постоянно быть в режиме подключения, ожидая запросов со стороны SMTP-клиента.

Логика протокола SMTP является действительно достаточно простой, как это и следует из его названия.

- После того как, применяя графический интерфейс своего почтового клиента, пользователь щелкает на значке отправки сообщения, SMTP-клиент посылает запрос на установление TCP-соединения на порт 25 SMTP-сервера (это назначенный порт).
- Если сервер готов, то он посылает свои идентификационные данные, в частности свое DNS-имя. Если SMTP-сервер оказался не готов, то он посылает соответствующее сообщение клиенту и тот снова посылает запрос, пытаясь заново установить соединение.
- Затем клиент передает серверу почтовые адреса (имена) отправителя и получателя.
- Если имя получателя соответствует ожидаемому, то после получения адресов сервер дает согласие на установление SMTP-соединения и в рамках этого логического канала происходит передача сообщения.
- Если после приема тела сообщения сервер отвечает командой OK, это означает, что сервер принял на себя ответственность по дальнейшей передаче сообщения получателю. Однако это не означает, что сервер гарантирует успешную доставку, потому что последнее зависит не только от него: например, клиентская машина получателя может быть в течение длительного времени не подсоединена к Интернету. Если сервер не может доставить сообщение, то он передает отчет об ошибке отправителю сообщения и разрывает соединение.
- Используя одно TCP-соединение, клиент может передать несколько сообщений, предваряя каждое из них указанием почтовых адресов отправителя и получателя.
- После завершения передачи сообщения TCP- и SMTP-соединения разрываются, и благополучно переданное сообщение сохраняется в буфере на сервере.

Нужно отметить, что в протоколе SMTP предусмотрены как положительные, так и отрицательные уведомления о доставке (промежуточной или окончательной) электронного письма. Однако только отрицательные уведомления являются обязательными, поэтому обычно SMTP-серверы предпочитают не передавать положительные уведомления в направлении отправителя.

ПРИМЕЧАНИЕ

Хотя в любом протоколе предполагается обмен данными между взаимодействующими частями, то есть данные передаются в обе стороны, различают протоколы, ориентированные на передачу (push protocols), и протоколы, ориентированные на прием данных (pull protocols). В протоколах, ориентированных на передачу, к которым, в частности, относится протокол SMTP, клиент является инициатором передачи данных на сервер, а в протоколах, ориентированных на прием, к которым относятся, например, протоколы HTTP, POP3 и IMAP, клиент является инициатором получения данных от сервера.

Непосредственное взаимодействие клиента и сервера

Теперь, когда мы обсудили основные составляющие почтовой службы, давайте рассмотрим несколько основных схем ее организации. Начнем с простейшего, практически неиспользуемого сейчас варианта, когда отправитель непосредственно взаимодействует с получателем. Как показано на рис. 23.3, у каждого пользователя на компьютере установлены почтовые клиент и сервер.

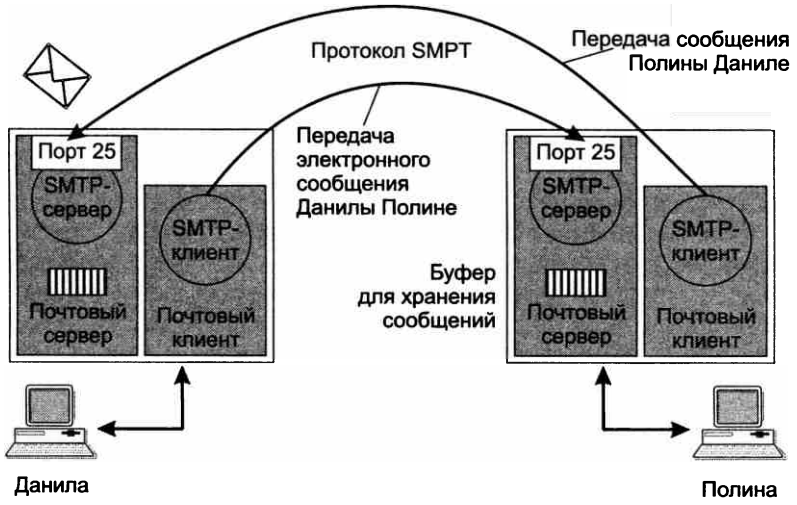


Рис. 23.3. Схема непосредственного взаимодействия клиента и сервера

Данила, используя графический интерфейс своего почтового клиента, вызывает функцию создания сообщения, в результате чего на экране появляется стандартная незаполненная форма сообщения, в поля которой Данила вписывает свой адрес, адрес Полины и тему письма, а затем набирает текст письма. При этом он может пользоваться не только встроенным в почтовую программу текстовым редактором, но и привлекать для этой цели другие программы, например MS Word. Когда письмо готово, Данила вызывает функцию отправки сообщения, и встроенный SMTP-клиент посылает запрос на установление связи SMTP-серверу на компьютере Полины. В результате устанавливаются SMTP- и TCP-соединения, после чего сообщение передается через сеть. Почтовый сервер Полины сохраняет письмо в памяти ее компьютера, а почтовый клиент по команде Полины выводит его на экран, при необходимости выполняя преобразование формата. Полина может сохранить, переадресовать или удалить это письмо. Понятно, что в случае, когда Полина решит направить электронное сообщение Даниле, схема работы почтовой службы будет симметричной.

Схема с выделенным почтовым сервером

Рассмотренная только что простейшая схема почтовой связи кажется работоспособной, однако у нее есть серьезный и очевидный дефект. Мы упоминали, что для обмена сообще-

ниями необходимо, чтобы SMTP-сервер постоянно находился в ожидании запроса от SMTP-клиента. Это означает, что для того, чтобы письма, направленные Полине, доходили до нее, ее компьютер должен постоянно находиться в режиме подключения. Понятно, что такое требование для большинства пользователей неприемлемо.

Естественным решением этой проблемы является размещение SMTP-сервера на специально выделенном для этой цели компьютере-посреднике. Это должен быть достаточно мощный и надежный компьютер, способный круглосуточно передавать почтовые сообщения от многих отправителей ко многим получателям. Обычно почтовые серверы поддерживаются крупными организациями для своих сотрудников или провайдерами для своих клиентов. Для каждого домена имен система DNS создает записи типа MX, в которых хранятся DNS-имена почтовых серверов, обслуживающих пользователей, относящихся к этому домену.

На рис. 23.4 представлена схема с выделенным почтовым сервером. Чтобы не усложнять рисунок, мы показали на нем только те компоненты, которые участвуют в передаче сообщения от Данилы к Полине. Для обратного случая схема должна быть симметрично дополнена.

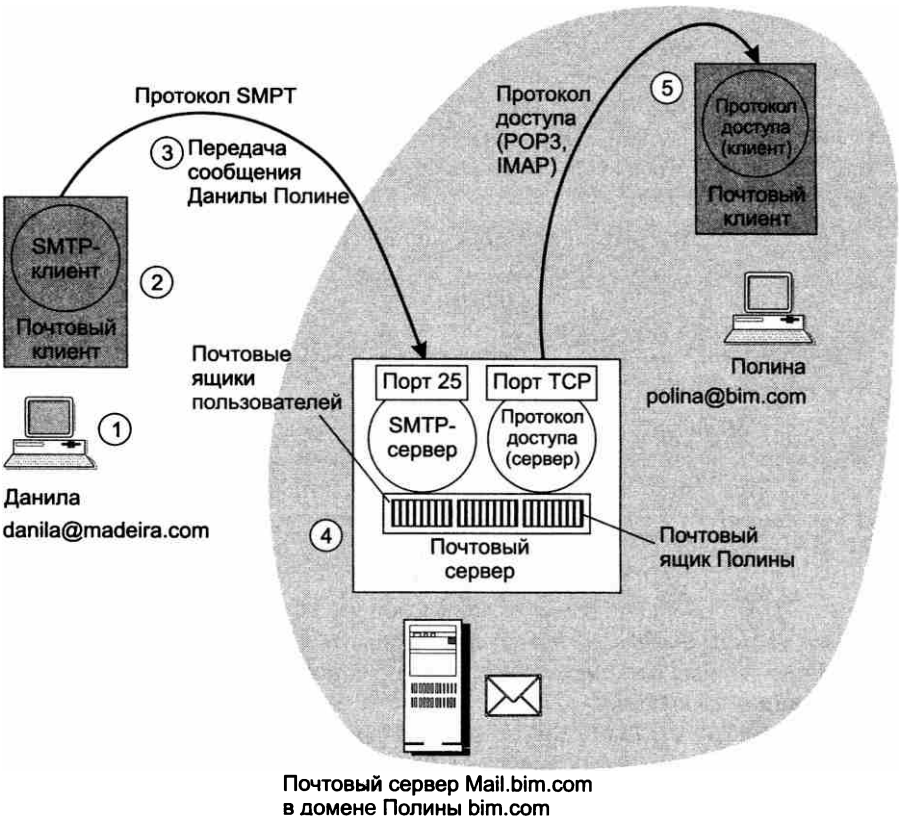


Рис. 23.4. Схема с выделенным почтовым сервером в принимающем домене

1. Итак, пусть Данила решает послать письмо Полине, для чего он запускает на своем компьютере установленную на нем программу почтового клиента (например, Microsoft Outlook или Mozilla Thunderbird). Он пишет текст сообщения, указывает необходимую сопроводительную информацию, в частности адрес получателя polina@bim.com, и щелкает мышью на значке отправки сообщения. Поскольку готовое сообщение должно быть направлено совершенно определенному почтовому серверу, клиент обращается к системе DNS, чтобы определить имя почтового сервера, обслуживающего домен Полины bim.com. Получив от DNS в качестве ответа имя mail.bim.com, SMTP-клиент еще раз обращается к DNS, на этот раз — чтобы узнать IP-адрес почтового сервера mail.bim.com.
2. SMTP-клиент посылает по данному IP-адресу запрос на установление TCP-соединения через порт 25 (SMTP-сервер).
3. С этого момента начинается диалог между клиентом и сервером по протоколу SMTP, с которым мы уже знакомы. Заметим, что здесь, как и у всех протоколов, ориентированных на передачу, направление передачи запроса от клиента на установление SMTP-соединения совпадает с направлением передачи сообщения. Если сервер оказывается готовым, то после установления TCP-соединения сообщение Данилы передается.
4. Письмо сохраняется в буфере почтового сервера, а затем направляется в индивидуальный буфер, отведенный системой для хранения корреспонденции Полины. Такого рода буферы называют почтовыми ящиками. Важно заметить, что помимо Полины у почтового сервера имеется еще много других клиентов, и это усложняет его работу. То есть почтовый сервер должен решать самые разнообразные задачи по организации многопользовательского доступа, включая управление разделяемыми ресурсами и обеспечение безопасного доступа.
5. В какой-то момент, который принципиально не связан с моментом поступления сообщений на почтовый сервер, Полина запускает свою почтовую программу и выполняет команду проверки почты. После этой команды почтовый клиент должен запустить протокол доступа к почтовому серверу. Однако это не будет SMTP. Напомним, что протокол SMTP используется тогда, когда необходимо передать данные на сервер, а Полине, напротив, нужно получить их с сервера.

Для этого случая были разработаны другие протоколы, обобщенно называемые протоколами доступа к почтовому серверу, такие, например, как **POP3** и **IMAP**. Оба этих протокола относятся к протоколам, ориентированным на прием данных. Инициатором передачи сообщений от почтового сервера почтовому клиенту по протоколу POP3 или IMAP является клиент. Почтовый сервер ожидает запрос на установление TCP-соединения по протоколу POP3 через порт 110, а по протоколу IMAP — через порт 143, на рисунке эти порты обобщенно изображены как порт TCP. В результате работы любого из них письмо Данилы оказывается в памяти компьютера Полины. Заметим, что на этот раз направление запроса от клиента к серверу не совпадает с направлением передачи данных, показанном стрелкой.

Оба протокола — POP3 и IMAP — поддерживают передачу отправителю квитанций, подтверждающих доставку, а также факт открытия сообщения получателем в случае, когда отправитель запрашивает такую почтовую услугу. Обычно для обеспечения приватности почтовый клиент не отправляет квитанцию автоматически, а спрашивает у получателя разрешение на такое действие.

Схема с двумя почтовыми серверами-посредниками

Прежде чем мы перейдем к сравнению двух протоколов доступа к почте, давайте рассмотрим еще одну схему организации почтовой службы, наиболее приближенную к реальности (рис. 23.5). Здесь передача сообщений между клиентами почты (на рисунке между отправителем Данилой и получателем Полиной) проходит через два промежуточных почтовых сервера, каждый из которых обслуживает домен своего клиента. На каждом из этих серверов установлены также и клиентские части протокола SMTP. При отправке письма почтовый клиент Данилы передает сообщение по протоколу SMTP почтовому серверу домена, к которому относится Данила, — RoyalMail.madeira.com. Это сообщение буферизуется на данном сервере, а затем по протоколу SMTP передается дальше на почтовый сервер домена Полины — mail.bim.com, откуда описанным уже образом попадает на компьютер Полины.

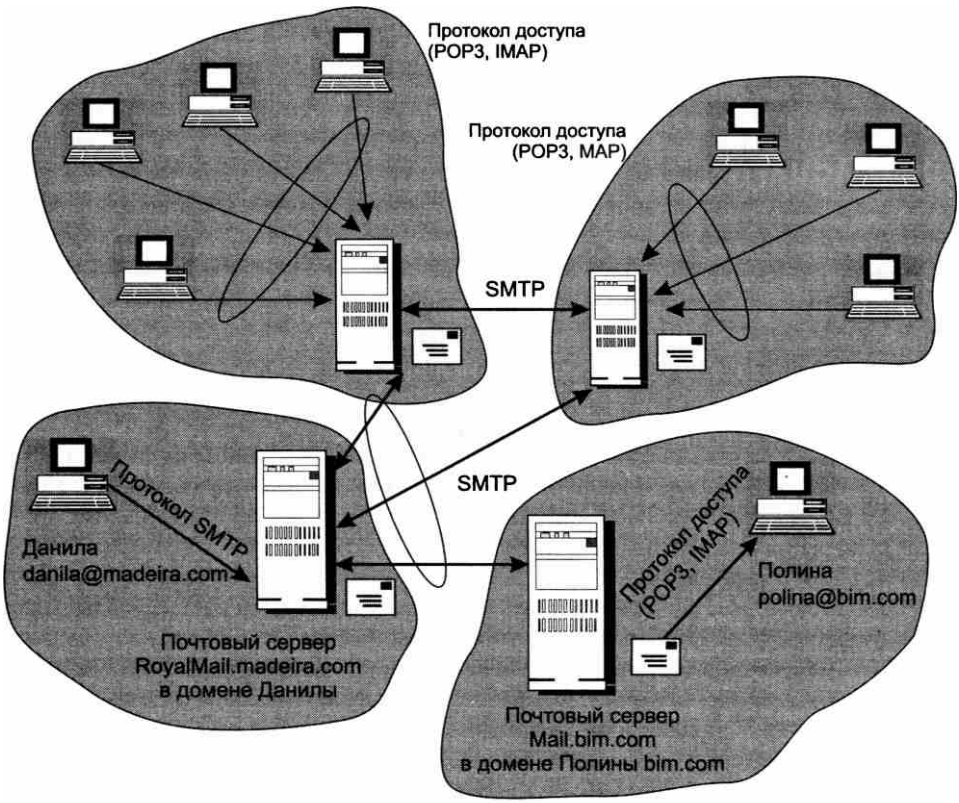


Рис. 23.5. Схема с выделенными почтовыми серверами в каждом домене

Возникает вопрос: зачем нужна такая двухступенчатая передача через два почтовых сервера? Прежде всего для повышения надежности и гибкости процедуры доставки сообщения. Действительно, в схеме с передачей сообщения сразу на сервер получателя почтовый

клиент отправителя в случае неисправности почтового сервера должен самостоятельно справляться со сложившейся нештатной ситуацией. Если же посредником в передаче сообщения является другой почтовый сервер, то это позволяет реализовывать разнообразные логические механизмы реакции на отказы на стороне сервера, который к тому же всегда находится в режиме подключения. Например, при невозможности передать письмо почтовому серверу получателя сервер отправляющей стороны может не только рапортовать об этом своему клиенту, но и предпринимать собственные действия — пытаться снова и снова послать письмо, повторяя эти попытки в течение достаточно длительного периода.

Протоколы POP3 и IMAP

А теперь сравним два протокола доступа к почте: **POP3** (*Post Office Protocol* v.3 — протокол почтового отделения версии 3) и **IMAP** (*Internet Mail Access Protocol* — протокол доступа к электронной почте Интернета). Оба протокола решают одну и ту же задачу — обеспечивают пользователей доступом к их корреспонденции, хранящейся на почтовом сервере. В связи с многопользовательским характером работы почтового сервера оба протокола поддерживают аутентификацию пользователей на основе идентификаторов и паролей пользователей. Однако протоколы POP3 и IMAP имеют и принципиальные различия, важнейшее из которых состоит в следующем. Получая доступ к почтовому серверу по протоколу POP3, вы «перекачиваете» адресованные вам сообщения в память своего компьютера, при этом на сервере не остается никакого следа от считанной вами почты. Если же доступ осуществляется по протоколу IMAP, то в память вашего компьютера передаются только копии сообщений, хранящихся на почтовом сервере.

Это различие серьезно влияет на характер работы с электронной почтой. Сейчас очень распространенной является ситуация, когда человек в течение одного и того же периода времени использует несколько разных компьютеров: на постоянном месте работы, дома, в командировке. Теперь давайте представим, что произойдет с корреспонденцией пользователя Полины, если она получает доступ к почте по протоколу POP3. Письма, прочитанные на работе, останутся в памяти ее рабочего компьютера. Придя домой, она уже не сможет прочитать их снова. Опросив почту дома, она получит все сообщения, которые поступили с момента последнего обращения к почтовому серверу, но из памяти сервера они исчезнут, и завтра на работе она, возможно, не обнаружит важных служебных сообщений, которые были загружены на диск ее домашнего ноутбука. Таким образом, получаемая Полиной корреспонденция «рассеивается» по всем компьютерам, которыми она пользовалась. Такой подход не позволяет рационально организовать почту: распределять письма по нескольким различным папкам, сортировать их по разным критериям, отслеживать состояние переписки, отмечать письма, на которые послан ответ, и письма, еще требующие ответа, и т. д. Конечно, если пользователь всегда работает только с одним компьютером, недостатки протокола POP3 не являются столь критичными. Но и в этом случае проявляется еще один «дефект» этого протокола — клиент не может пропустить, не читая, ни одного письма, поступающего от сервера. То есть объемное и, возможно, совсем ненужное вам сообщение может надолго заблокировать вашу почту.

Протокол IMAP был разработан как ответ на эти проблемы. Предположим, что теперь Полина получает почту по протоколу IMAP. С какого компьютера она бы ни обратилась к почтовому серверу, ей будут переданы только копии запрошенных сообщений. Вся совокупность полученной корреспонденции останется в полной сохранности в памяти

почтового сервера (если, конечно, не поступит специальной команды от пользователя об удалении того или иного письма). Такая схема доступа делает возможным для сервера предоставление широкого перечня услуг по рациональному ведению корреспонденции, то есть именно того, чего лишен пользователь при применении протокола POP3. Важным преимуществом IMAP является также возможность предварительного чтения заголовка письма, после чего пользователь может принять решение о том, есть ли смысл получать с почтового сервера само письмо.

IP-телефония

IP-телефония — это сетевая служба, предоставляющая телефонные услуги передачи голоса по IP-сети.

Понятие «IP-телефония» распространяется также и на те случаи, когда голос и факс передаются вместе с другими видами информации, в частности с текстом и изображением. Помимо термина «IP-телефония» употребляются также термины «VoIP» (Voice over IP — голос через IP) и «интернет-телефония». Хотя аббревиатура VoIP часто используется как синоним термина «IP-телефония», существует ее более широкая трактовка — любая услуга, включающая передачу голоса по протоколу IP; это может быть, например, передача голосовой рекламы при щелчке на соответствующем значке, расположенном на веб-странице. Интернет-телефония — это частный случай IP-телефонии, когда разговор происходит через Интернет, а не, например, в пределах IP-сети предприятия.

Ранняя IP-телефония

В своем развитии IP-телефония прошла три этапа.

На *первом этапе* это была скорее интернет-игрушка, пригодная разве что для общения двух энтузиастов, готовых мириться с сопровождающим диалог кваканьем и шипением. Два компьютера, оснащенные микрофонами, динамиками, звуковыми картами с поддержкой оцифровки звука и не очень сложным программным обеспечением, позволяли вести двусторонний диалог через Интернет в реальном времени (рис. 23.6).

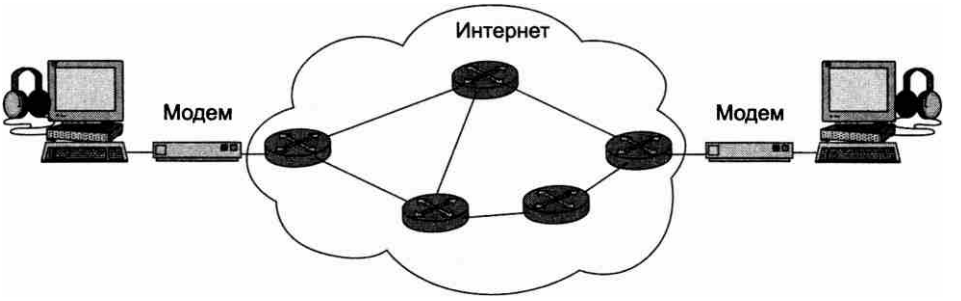


Рис. 23.6. Средства поддержки разговора пользователей через Интернет

Однако до удобств обычной телефонной услуги такой способ общения явно недотягивал. Абонентам нужно было знать IP-адрес компьютера собеседника, договариваться о времени разговора, выбирать момент для более качественной передачи речи, когда трафик Интернета между данными конкретными точками не сталкивался с перегрузками и задержками. Кроме того, при отсутствии стандартов на обоих компьютерах требовалось установить такое программное обеспечение, которое поддерживало бы один и тот же способ кодирования голоса и упаковки его в пакеты. Взаимодействия между компьютером и телефоном, подключенным к обычной телефонной сети, не предполагалось. Зато затраты ограничивались небольшой платой провайдеру за обычное коммутируемое подсоединение к Интернету.

Второй этап ознаменовался появлением стандартов IP-телефонии, прежде всего стандартов группы H.323, разработанных ITU-T, и стандартов на основе протокола SIP, разработанных IETF.

К *третьему этапу* можно отнести появление нового поколения IP-телефонии, поддерживающей широкий спектр дополнительных услуг, подобный тому, который предоставляют абонентам развитые телефонные сети.

Стандарты H.323

Разработчики **стандартов H.323** исходили из того, что две сети — телефонная и IP — будут сосуществовать бок о бок достаточно длительное время, а значит, важно регламентировать их взаимодействие с учетом существующих в традиционных телефонных сетях процедур установления соединения, а также договориться о способе передачи вызова и собственно голоса по IP-сети.

В рамках установленного сеанса H.323 абоненты могут обмениваться не только голосовой информацией, но и видеоинформацией, то есть пользоваться видеотелефонами или оборудованием для организации видеоконференций.

В стандартах H.323 определяется две группы протоколов (рис. 23.7).

Протоколы транспортной (transport plane), или *пользовательской* (user plane), *плоскости* отвечают за непосредственную передачу голоса по сети с коммутацией пакетов. Протоколы этой плоскости определяют способы кодирования голоса (сюда входят стандарты различных кодеков, например G.711, G.723.1, G.729, G.728 и др.) и видео (кодеки H.261, H.263 и др.). Голос и видео передаются в пакетах протокола RTP (Real Time Protocol — протокол реального времени), который определен в RFC 3550 и переносит отметки времени и последовательные номера пакетов, помогая конечным узлам сеанса восстанавливать аналоговую информацию реального времени. RTP-пакеты переносятся в пакетах протокола UDP.

Протоколы плоскости управления вызовами (call control plane) переносят по сети запросы на установление соединений и реализуют такие служебные функции, как авторизация доступа абонента к сети и учет времени соединения. Эта группа протоколов работает через надежные TCP-соединения и включает протокол сигнализации Q.931, обеспечивающий установление и завершение соединения между абонентами; протокол H.245, с помощью которого абонентское оборудование узнает о функциональных возможностях противоположной стороны, например о том, какие аудио- и видеокодеки поддерживаются, а также о том, сколько аудио- и видеопотоков будут использовать абоненты в рамках данного соединения. По умолчанию IP-телефон поддерживает только один голосовой поток, но ви-

деотелефон уже поддерживает два потока — один голосовой и один видео, а оборудование видеоконференции может поддерживать несколько аудиопотоков и несколько видеопотоков. Еще один протокол этой группы — RAS (Registration, Admission, Status) — служит для учета звонков, регистрации пользователя в некотором административном домене (например, в домене организации, где работает пользователь) и контроля доступа в сеть. Контроль доступа в данном случае заключается в проверке наличия сетевых ресурсов, необходимых для качественного обслуживания телефонного вызова, например свободной пропускной способности.

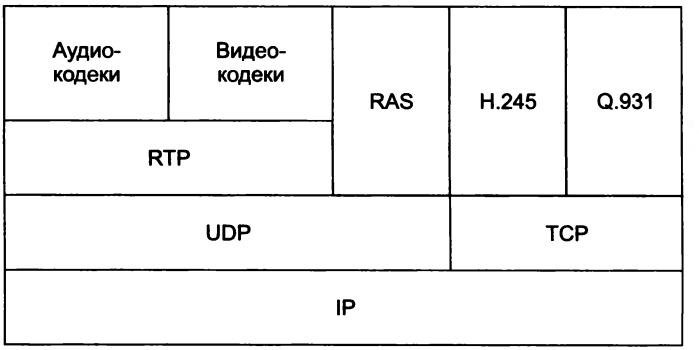


Рис. 23.7. Стек протоколов H.323

Основными элементами сети H.323, в которых реализуются протоколы этого стека, являются IP-телефоны, шлюзы и привратники (рис. 23.8).

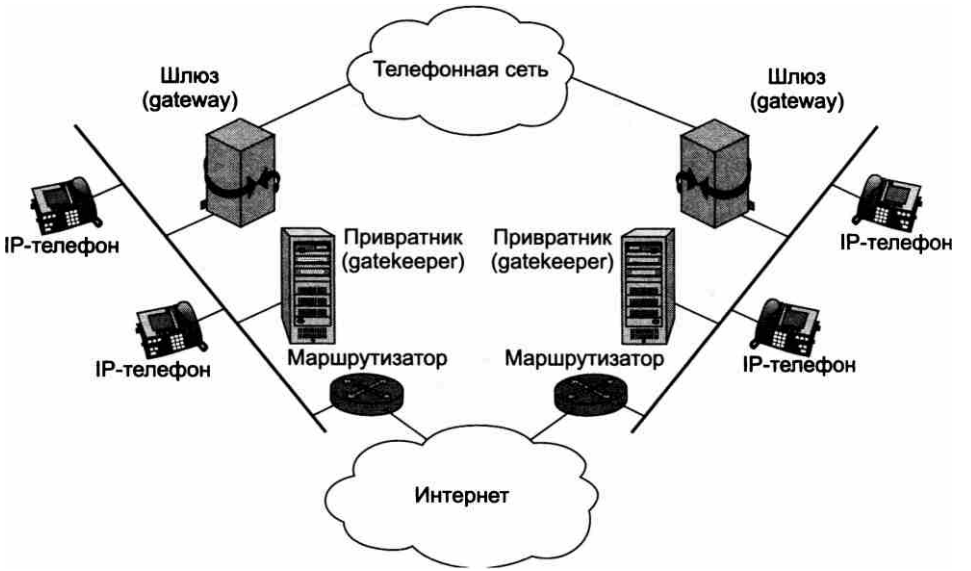


Рис. 23.8. Элементы сети H.323

IP-телефон представляет собой обычный телефонный аппарат с кнопочным набором и небольшим дисплеем для отображения информации о вызовах и справочной информации (например, телефонных номеров абонентов предприятия). В отличие от обычного телефона, IP-телефон подключается непосредственно к IP-сети через соединение Ethernet.

Шлюз (gateway) связывает традиционную телефонную сеть с IP-сетью, он обеспечивает трансляцию упакованного в пакеты оцифрованного и зачастую сжатого голоса в форму, пригодную для передачи по телефонной сети общего пользования. Кроме того, в функции шлюза H.323 входит трансляция протоколов сигнализации телефонных сетей, таких, например, как SS7, в протоколы сигнализации стека H.323. Шлюз позволяет абонентам с обычным телефонным аппаратом общаться с пользователями IP-телефонов или же задействовать IP-сеть как транзитную.

Основная задача плоскости управления вызовами — установление соединения между абонентами через сети с коммутацией пакетов — в простейшем случае может быть решена шлюзом, а в более общей постановке поручается специальному сетевому объекту — привратнику.

Привратник (gatekeeper) выполняет регистрацию и авторизацию абонентов по протоколу RAS, а также, в случае необходимости, трансляцию адресов (например, DNS-имен в телефонные номера). Кроме того, он занимается маршрутизацией вызовов к IP-телефону или шлюзу, а если потребуется, то и к другому привратнику.

Обычно один привратник обслуживает так называемую зону, то есть часть сети, находящуюся под административным управлением одной организации. Все функции привратника в архитектуре H.323 могут выполнять терминальные устройства — телефоны и шлюзы, но такое решение плохо масштабируется, а поток вызовов с трудом контролируется и тарифицируется.

> В функции привратника может также входить взаимодействие с интеллектуальной сетью. Интеллектуальная сеть (Intelligent Network, IN) является частью современной цифровой телефонной сети, которая предоставляет ее абонентам дополнительные услуги, такие как переадресация вызова, конференц-связь, телеголосование и др. Интеллектуальная сеть по своей сути является компьютерной сетью с серверами, на которых программируется логика услуг.

Стандарты на основе протокола SIP

Основным конкурентом протоколов стандарта H.323 является протокол **SIP** (Session Initiation Protocol — протокол инициирования сеанса), разработанный интернет-сообществом и стандартизованный IETF в RFC 3261. SIP является сигнальным протоколом, он ответствен за установление сеанса между абонентами, при этом SIP выполняет функции протоколов Q.931, RAS и H.245 стандарта H.323 (точнее — часть из них). Для передачи аудио- и видеоданных в ходе сеанса протокол SIP предполагает использование протокола RTP.

Протокол SIP очень близок по стилю к протоколу HTTP: он имеет похожий набор и синтаксис сообщений, которыми обмениваются стороны в процессе установления сеанса. Как и у протокола HTTP, SIP-сообщения текстовые, они вполне понятны программистам, имеющим опыт создания веб-приложений. Поэтому системы IP-телефонии, построенные на

основе SIP, оказались гораздо ближе к миру Интернета, чем стандарты H.323, пришедшие «от телефонистов». Сегодня SIP-телефония более тесно интегрирована с веб-услугами, чем телефония стандарта H.323.

Архитектура SIP предусматривает как непосредственное взаимодействие абонентов через IP-сеть, так и более масштабируемые схемы, включающие участие серверов-посредников. Основным таким сервером является так называемый **прокси-сервер SIP**, он выполняет функции, близкие к функциям привратника H.323. Кроме того, в архитектуре SIP может присутствовать **сервер определения местоположения** (SIP Location Server).

Работу протокола SIP в архитектуре с серверами обоего типа иллюстрирует рис. 23.9.

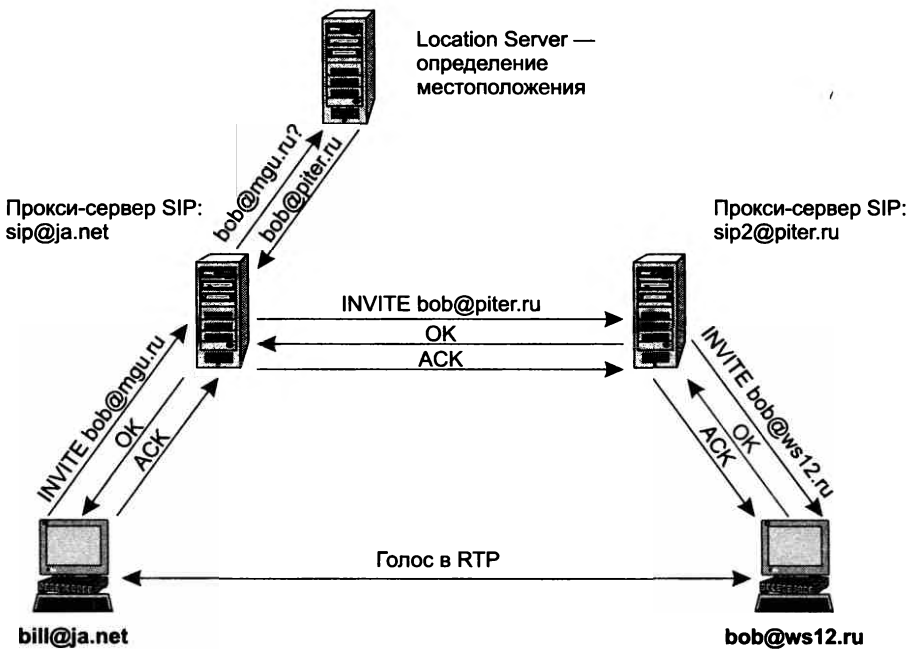


Рис. 23.9. Взаимодействие абонентов SIP

Адресами абонентов в протоколе SIP являются универсальные идентификаторы (URI), используемые во всех веб-службах. На рис. 23.9 абонент bill@ja.net хочет установить сеанс с абонентом bob@mgu.ru. В домене ja.net установлен прокси-сервер SIP с именем sip1@ja.net, через него проходят все вызовы абонентов этого домена (за счет того, что в IP-телефонах абонентов задан IP-адрес этого прокси-сервера).

Запросом на установление сеанса в протоколе SIP является передача сообщения *INVITE* с URI вызываемого абонента, поэтому абонент bill@ja.net направляет своему прокси-серверу сообщение *INVITE bob@mgu.ru*. Прокси-сервер для выполнения этого запроса обращается к серверу определения местоположения, который возвращает ему ответ о том, что абонент bob@mgu.ru в данный момент зарегистрирован как активный в домене piter.ru с именем bob@piter.ru. Прокси-сервер использует эту информацию для того, чтобы направить сообщение *INVITE* прокси-серверу домена piter.ru (сервер с именем sip2@piter.ru), указав в нем

имя bob@piter.ru. Вызов завершается прокси-сервером sip2@piter.ru, который обнаруживает, что пользователь bob@piter.ru зарегистрировался и работает в настоящее время за компьютером ws12, поэтому вызов *INVITE* передается на этот компьютер. Далее протокол SIP работает подобно большинству протоколов сигнализации: если пользователь bob@ws12.ru соглашается принять вызов, то он снимает трубку своего SIP-телефона (или щелкает на соответствующем значке своего программного SIP-телефона) и тем самым посылает ответ *OK* назад по цепочке. Окончательное установление сеанса фиксируется отправкой сообщения *ACK* (подтверждение) от вызывающего абонента к вызываемому.

После установления сеанса разговор происходит между телефонами абонентов в рамках протокола RTP.

Существуют также фирменные протоколы IP-телефонии, из которых наиболее известными являются **протоколы Skype** — очень популярного сервиса интернет-телефонии. Этот сервис к тому же поддерживает такие дополнительные услуги, как видеоконференции, передача мгновенных сообщений, передача файлов между абонентами.

Связь телефонных сетей через Интернет

На втором этапе развития IP-телефонии IP-сеть (Интернет или частная сеть) широко использовалась в качестве транзитной сети между двумя местными телефонными сетями (рис. 23.10). Данная схема реализации общедоступных услуг IP-телефонии стала достаточно популярной во всем мире, в том числе и в России. Она заключается в том, что абонент звонит по определенному номеру, который закреплен за провайдером местной телефонной сети, и на звонок отвечает **сервер интерактивного голосового ответа** (Interactive Voice Response, IVR). IVR-сервер запрограммирован на выполнение рутинных процедур аутентификации вызывающего абонента и приема номера вызываемого абонента. Для этого привлекается техника распознавания голосовых ответов (которыми могут быть и сигналы тонового набора, используемого вызывающим абонентом для ответов на запросы IVR-сервера).

Для реализации услуги IP-телефонии по описанной схеме оператору связи не надо создавать собственную дорогостоящую транспортную инфраструктуру и иметь непосредственный доступ к абонентам. Однако стратегические перспективы такого подхода оставляют желать лучшего из-за плохой масштабируемости и узкого спектра услуг.

Масштабируемость описанного варианта ограничивается несколькими факторами. Во-первых, провайдеру приходится устанавливать многочисленные одноранговые связи со своими друзьями-соперниками по бизнесу. Во-вторых, протоколы обеих плоскостей необходимо реализовывать во всех элементах сети IP-телефонии: и в привратниках, и в шлюзах, и в терминалах, что приводит к излишней сложности и дороговизне всех этих устройств. И наконец, пользователям предоставляются только базовые услуги по обработке вызовов, поскольку взаимодействие с такими протоколами телефонной сети, как протокол межстанционной сигнализации (SS7) и протоколы интеллектуальной сети (IN), отсутствует. Эту последнюю группу недостатков нельзя отнести на счет стандартов H.323, в которых явно не говорится о том, какие сигнальные протоколы должен поддерживать шлюз со стороны телефонной сети. Перечень дополнительных услуг по обработке вызовов определен в спецификации H.450. Таким образом, это скорее изъян реализации шлюзов того поколения, в которых поддержка SS7 и IN, как правило, отсутствовала.

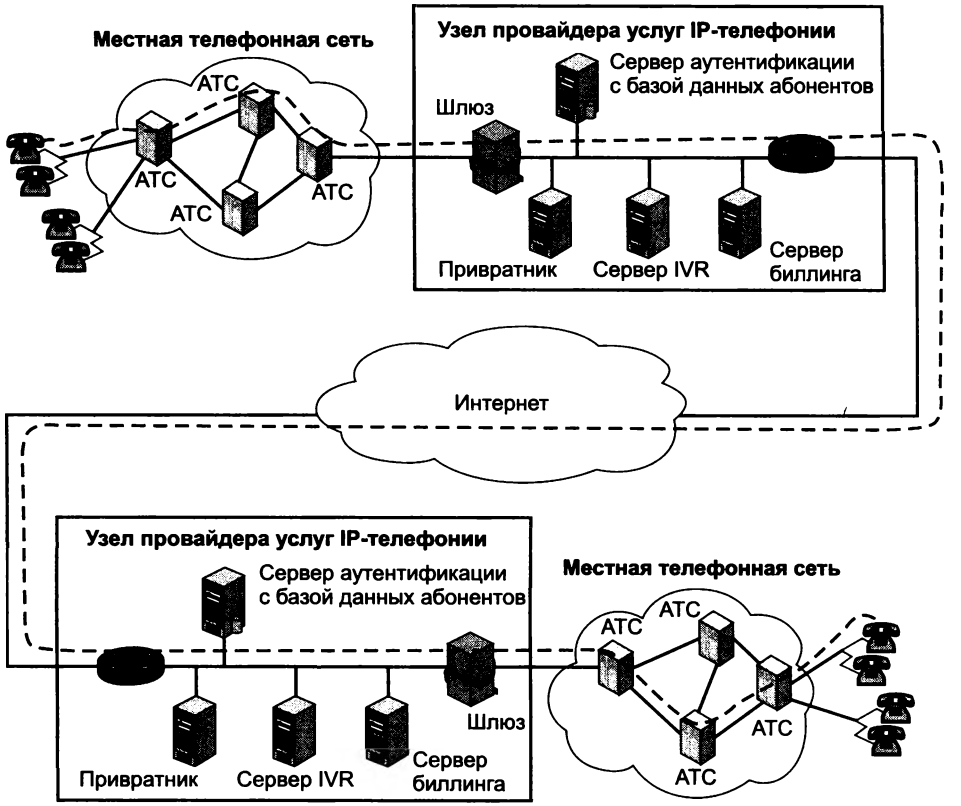


Рис. 23.10. Взаимодействие двух местных телефонных сетей через Интернет

Кроме того, сам диалог достаточно утомителен — гораздо удобнее просто набрать номер с небольшой приставкой вроде 8-20 и получить доступ к услугам международной IP-телефонии. Но для этого провайдеру нужен прямой доступ к абоненту или договоренность с местными операторами о переадресации таких вызовов на шлюз IP-телефонии провайдера с помощью средств интеллектуальной сети (а они пока поддерживаются далеко не всеми местными операторами). Таким образом, для выхода IP-телефонии на более высокий уровень национального или международного оператора требуются другие стандарты и оборудование, чтобы сети, построенные на базе протокола IP, могли равноправно соседствовать с традиционными телефонными сетями.

Многие из необходимых стандартов уже появились и воплощены в новом поколении оборудования, ставшем основой для *третьего этапа* развития IP-телефонии.

Третье поколение сетей IP-телефонии

Укрупненная схема полномасштабной сети IP-телефонии показана на рис. 23.11. Такая сеть может поддерживать собственных абонентов и служить транзитной для традиционных телефонных сетей с оказанием полного спектра услуг, включая услуги интеллектуальной сети.

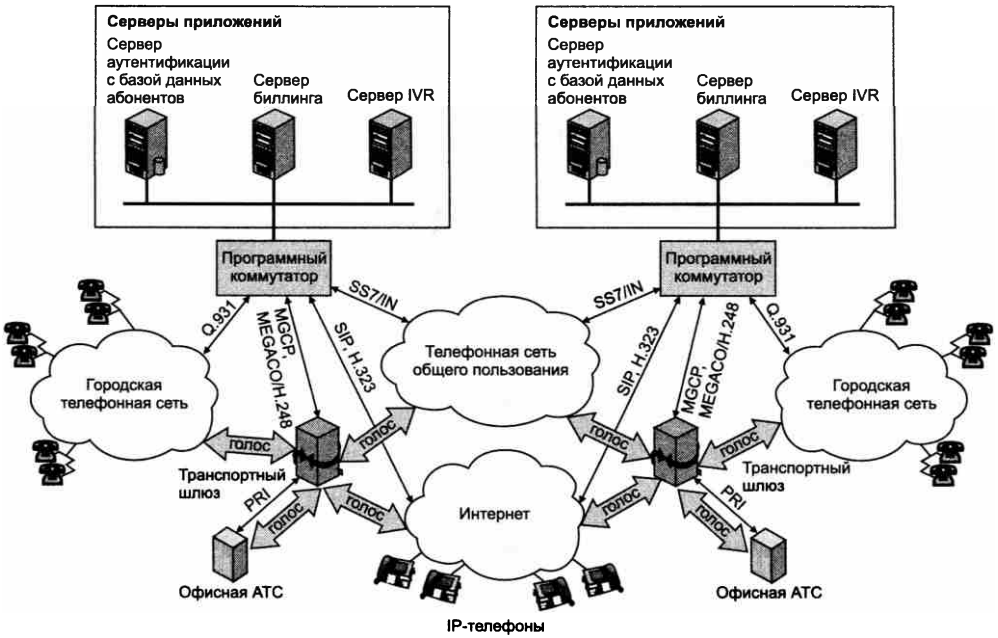


Рис. 23.11. Масштабируемая архитектура IP-телефонии

Эта сеть обладает несколькими отличительными особенностями. Так, в узлах IP-телефонии нового поколения произошло четкое разделение функций на три группы:

- транспортную;
- управления вызовами;
- прикладных сервисов.

Транспортная группа образовалась за счет выделения из шлюза функциональной части, выполняющей очень простую операцию — коммутацию между входными и выходными портами (физическими или виртуальными). Этот элемент, получивший название **транспортного шлюза** (Media Gateway, MG), является своего рода аналогом коммутационного поля телефонной станции.

Следующую группу — группу управления вызовами — составляют протоколы сигнализации IP-телефонии (H.225.0, RAS из стандарта H.323 или SIP). К этой группе относят также протоколы управления транспортными шлюзами, которые инициируют действия по коммутации портов. Все перечисленные базовые функции по обработке вызовов сегодня часто реализуются одним устройством — так называемым **программным коммутатором** (softswitch).

Третья группа функций образует уровень сервисов, реализуемых в виде обычных сетевых приложений универсальными серверами. Примерами таких сервисов являются инициирование телефонного вызова при щелчке на определенной кнопке веб-страницы, передача вызова абоненту, подключенному к Интернету по телефонной сети, а также услуги интеллектуальной сети.

В сетях IP-телефонии второго этапа развития уровень сервисов практически отсутствовал — пользовательские услуги оказывал только IVR-сервер, а остальные прикладные программные системы этого уровня реализовывали внутренние для провайдера функции — аутентификацию, биллинг и т. п. Теперь уровень сервисов поддерживает весь спектр дополнительных услуг, которые могут предоставлять абонентам развитые телефонные коммутаторы городского типа, в том числе и с помощью интеллектуальной сети: переадресацию вызовов в соответствии с различными условиями, телеголосование, бесплатный звонок, звонок по специальному тарифу, сокращенный набор и т. п.

Очень важно, что взаимодействие между уровнями осуществляется через стандартные интерфейсы, а это создает серьезные предпосылки для построения телефонных узлов IP-телефонии на основе продуктов разных производителей с применением общепринятых способов обработки вызовов. Такой унифицированный модульный подход был бы очень привлекателен и при разработке традиционных телефонных сетей, однако производители телефонных коммутаторов обычно реализовывали функции двух нижних уровней и взаимодействие между ними с использованием собственных корпоративных стандартов. Только при создании архитектуры интеллектуальной сети удалось, наконец, воплотить в жизнь принцип независимости верхнего уровня от двух нижних и принять в качестве **стандарта межуровневого взаимодействия** протокол **INAP** (Intelligent Network Application Protocol — прикладной протокол интеллектуальной сети), работающий поверх протоколов системы сигнализации SS7.

Распределенные шлюзы и программные коммутаторы

Масштабируемость коммутации и независимость транспортного уровня от уровня управления вызовами в новом поколении узлов IP-телефонии достигается благодаря применению концепции программного коммутатора. Сам термин «softswitch» получил широкое распространение в названиях продуктов, компаний и неформальных объединений. Данный управляющий элемент отвечает за обработку сообщений сигнальных протоколов, на основании которых происходят соединения: например, протокола H.225.0 стека H.323, протокола установления соединений SIP или же сигнального протокола SS7.

С помощью специального протокола «главный-подчиненный» программный коммутатор управляет транспортными шлюзами, которые в конечном счете и осуществляют коммутацию голосовых каналов. Для управления шлюзами сегодня могут использоваться несколько близких по логике работы протоколов: **SGCP** (Simple Gateway Control Protocol), **MGCP** (Media Gateway Control Protocol) или **MEGACO/H.248**. Собственно, стандартом, принятым как IETF, так и ITU-T, является только совместно разработанный ими протокол MEGACO/H.248, однако и предшественники этого стандарта, протоколы SGCP и MGCP, успешно реализуются в продуктах различных производителей. С помощью одного из названных протоколов программный коммутатор выясняет детали текущего состояния соединений и портов шлюза, а также передает ему указания о том, какую пару портов (физических или логических) требуется соединить, и некоторые другие предписания. Таким образом, реализация шлюза может быть весьма простой, а весь интеллект управления соединениями перемещается на уровень программного коммутатора, который в модели

распределенной коммутации управляет одновременно несколькими шлюзами. Именно такой вариант показан на рис. 23.11.

В протоколах SGCP, MGCP и MEGACO/H.248 управляющий элемент называется **агентом вызова** (call agent), однако программный коммутатор — это нечто большее, чем агент управления вызовами. Обычно в продукт с маркой softswitch производители помещают элементы уровня управления вызовами нескольких стандартов, чтобы такой программный коммутатор мог взаимодействовать с другими зонами телефонной сети по наиболее популярным сигнальным протоколам. Так, в программный коммутатор может входить привратник стандарта H.323, серверы стандарта SIP (прокси-сервер, сервер переадресации и сервер определения местоположения пользователей), а также шлюзы телефонной сигнализации для преобразования протоколов телефонных сетей в сигнальные протоколы IP-телефонии — те же SIP и H.225.0 стека H.323. Широкая поддержка сигнальных протоколов позволяет программному коммутатору находить общий язык практически с любыми типами телефонных сетей, как с традиционными (с коммутацией каналов), так и с пакетными.

Программные коммутаторы — «сердце» современного узла IP-телефонии — осуществляют за единицу времени множество соединений, столько же, сколько телефонные коммутаторы городского и междугородного типов. Высокая степень масштабируемости достигается благодаря распределенной модели коммутации, элементы которой взаимодействуют стандартным образом, что обеспечивает модульное построение узла коммутации.

Новые услуги

В промежуточных устройствах IP-сети не хранится информация о каждом соединении абонентов (компьютеров пользователей) с серверами. Это одно из принципиальных отличий IP-сети от телефонной сети. Коммутаторы телефонной сети, напротив, отслеживают и запоминают состояние каждого вызова, что является одной из причин более высокой стоимости передачи через них транзитного трафика по сравнению с IP-маршрутизаторами.

В публикациях по IP-телефонии постоянно подчеркивается, что удешевление звонков и оказание конкурентного давления на сектор традиционной международной телефонии — это краткосрочное преимущество IP-телефонии. Что же касается дальней стратегической перспективы, то основным направлением здесь будет предоставление новых услуг, в том числе интегрированных с услугами по передаче данных и манипулированию данными. К ним относятся:

- Click to Talk — инициирование телефонного разговора при просмотре веб-страницы;
- Internet Call Waiting (ICW) — уведомление абонента, подключившегося с помощью телефонной сети к Интернету, о наличии входящего вызова и, возможно, организация параллельного с интернет-сеансом разговора путем пакетной передачи;
- Unified Messaging — организация единой почтовой службы для любых сообщений, в том числе электронной почты, факсов и голоса, с возможностью трансформации вида представления информации.

Разнообразие услуг, их настройка в соответствии с потребностями конкретного пользователя, простота программирования нового предложения, легкость интеграции голосовых услуг с услугами манипулирования данными — это «врожденные» сильные стороны IP-телефонии, ее стратегический потенциал. Часть этих услуг, описываемых стандартами

SIP и H.245 как дополнительные, может предоставлять непосредственно программный коммутатор, более сложные сервисы реализуются с помощью серверов приложений узла IP-телефонии.

Интеграция систем адресации E.164 и DNS на основе ENUM

Одной из проблем современной IP-телефонии является сложность установления соединения, когда инициировавший вызов абонент использует обычный телефонный аппарат, подключенный к традиционной телефонной сети, а вызываемый абонент — компьютер или IP-телефон, соединенный с Интернетом или частной IP-сетью. Сложность подобного соединения связана с применением в общедоступных телефонных сетях и в Интернете *разных схем адресации* — системы телефонных номеров на основе международного стандарта E.164 и системы DNS-имен. И если пользователю компьютера или цифрового IP-телефона не составляет труда набрать телефонный номер для вызова абонента, то представить себе набор DNS-имени с помощью обычного аналогового аппарата довольно сложно.

Для преодоления пропасти между этими видами общедоступных услуг необходимо либо выбрать единую схему идентификации абонентов, либо разработать метод трансляции одной схемы в другую. Предложения **ENUM** (E.164 NUmber Mapping — отображение адресов стандарта E.164) рабочей группы IETF решают задачу вторым способом, и пока этот вариант наиболее близок к немедленной реализации. Подход ENUM, описанный в RFC 3761, состоит в назначении всем абонентам IP-телефонии, подключенным к Интернету или частной IP-сети, идентификаторов еще одного типа — телефонных номеров стандарта E.164. Однако на конечных узлах и даже сетях, в которых вызов терминируется, эти телефонные номера не используются — они нужны только для идентификации вызываемого абонента стороной-инициатором, применяющей обычный телефон, и маршрутизации вызова в пределах традиционной телефонной сети. Затем телефонные номера преобразуются в имена Интернета с помощью хорошо известной и отлично зарекомендовавшей себя службы — системы доменных имен (DNS).

Используемый при этом подход подобен тому, который применяется для решения обратной задачи — нахождение имени узла по его IP-адресу. С этой целью предлагается создать новую зону e164.arpa, куда будут входить территории, соответствующие цифрам телефонного номера, например зоны верхнего уровня 1, 7, 33, 44 для номеров, принадлежащих абонентам Североамериканского региона, России, Франции и Великобритании соответственно. Домен верхнего уровня arpa традиционно отводится для решения обратной задачи — нахождение имени по адресу с помощью зоны in-addr.arpa.

Для преобразования телефонного номера в DNS-имя используется специальный тип записи — Naming Athority Pointer (NAPTR). Изначально данная запись предназначалась для перечисления сервисов, которые поддерживает организация, администрирующая данный домен (RFC 2915). Примером такой записи может служить строка sip:Petrov@firma.ru, сообщающая о том, что с абонентом можно связаться, направив ему вызов по протоколу SIP на имя Petrov@firma.ru. Очевидно, что такие записи будут находиться только в зонах самого нижнего уровня, где располагается база номеров, которую провайдер получил для обслуживания конечных абонентов. Зоны же верхнего уровня будут содержать только обычные ссылки на серверы имен зон более низкого уровня. Итак, если имени Petrov@firma.ru соответствует телефонный номер +7 095 758 35 22, то связанная с этим абонентом запись, воз-

можно, содержится в зоне 8.5.7.5.9.0.7.e164.arpa (обратный порядок записи цифр телефонного номера согласуется с принятым в DNS правилом расположения старшей части имени справа, а не слева, как в телефонии). Запись может находиться и в зоне 3.8.5.7.5.9.0.7.e164.arpa, если все номера диапазона +7 095 758 3*x xx* переданы еще более мелкому провайдеру (в предыдущем примере предполагалось, что все номера +7 095 758 *xx xx* принадлежали одному провайдеру). Деление телефонного номера на зоны производится по цифрам в полном соответствии с административной ответственностью каждой конкретной организации за отображение телефонных номеров на DNS-имена (точнее, на URL-адреса, которые в дополнение к DNS-имени имеют префикс, указывающий на протокол доступа к ресурсу). Чем больше уровней подчиненности провайдеров IP-телефонии, тем больше составных компонентов в имени зоны.

Выводы

С точки зрения пользователей, компьютерные сети представляют собой набор служб (сервисов), таких как электронная почта, WWW, интернет-телефония и интернет-телевидение.

Важнейшей сетевой службой является World Wide Web (WWW), или Всемирная паутина; благодаря которой люди получили возможность доступа к огромному объему информации в удобном для них виде и в удобное для них время.

Клиентская часть веб-службы, называемая также браузером, представляет собой приложение, которое устанавливается на компьютере конечного пользователя и одной из важных функций которого является поддержание графического пользовательского интерфейса.

Веб-сервер — это программа, хранящая объекты локально в каталогах компьютера, на котором она запущена, и обеспечивающая доступ к этим объектам по URL-адресам.

Клиент и сервер веб-службы связываются через сеть по протоколу передачи гипертекста HTTP.

Электронная почта — это распределенное приложение, которое построено в архитектуре клиент-сервер и главной функцией которого является предоставление пользователям сети возможности обмениваться электронными сообщениями. Почтовый клиент и почтовый сервер применяют в своей работе специально разработанные для почтовых систем протоколы SMTP, POP3 и IMAP.

IP-телефония — это сервис, который обеспечивает коммутируемые голосовые соединения преимущественно по схеме «один к одному» и поддерживается сетью, использующей протокол IP, в форме общедоступного Интернета или частной IP-сети.

Важнейшим событием в IP-телефонии стало появление стандартов группы H.323, разработанных ITU-T, и стандартов на основе протокола SIP, разработанных IETF.

Новое поколение IP-телефонии поддерживает широкий спектр услуг, подобный тому, который предоставляют абонентам развитые телефонные сети.

Контрольные вопросы

1. Известно, что единственным идентификатором получателя электронной почты, в том числе в схеме с выделенным почтовым сервером, является адрес пользователя вида

name@domain.com. Каким образом письмо находит путь к почтовому серверу, обслуживающему данного получателя?

2. Заполните таблицу, описывающую свойства почтовых протоколов IMAP, POP3 и SMTP:

Свойство протокола	**Протоколы**
Используется почтовым клиентом для передачи письма на сервер	
Используется почтовым клиентом для получения письма с сервера	
При получении почты письмо перемещается с сервера на клиент	
При получении почты письмо копируется с сервера на клиент	

3. Браузер находит информацию по адресам специального формата, например такому: http://www.bbc.co.uk/mobile/web/versions.shtml. Поместите в правый столбец таблицы части приведенного адреса, соответствующие названиям в левом столбце:

Путь к объекту	
DNS-имя сервера	
URL-имя	
Тип протокола доступа	

4. Что вы можете сказать о HTTP-сообщении вида HTTP/1.1 200 OK? Варианты ответов:
   а) это HTTP-запрос;
   б) это HTTP-ответ;
   в) 200 — это код состояния;
   г) 200 — это объем переданной информации;
   д) OK означает, что информация зашифрована открытым ключом;
   е) OK означает, что «все в порядке!».
5. Что вы можете сказать о протоколе SIP? Варианты ответов:
   а) это протокол веб-службы;
   б) это протокол IP-телефонии;
   в) входит в семейство протоколов H.323;
   г) похож на протокол HTTP;
   д) выполняет примерно те же функции, что и протоколы Q.931, RAS и H.245.

ГЛАВА 24 Сетевая файловая служба

Элементы сетевой файловой службы

Сетевая файловая служба, или **система (ФС)**, предоставляет пользователям сети услуги по совместному использованию файлов, хранящихся на компьютерах сети.

Сетевая ФС в общем случае включает следующие элементы (рис. 24.1):

- клиент сетевой ФС;
- сервер сетевой ФС;
- интерфейс сетевой ФС;
- локальная ФС;
- интерфейс локальной ФС;
- протокол клиент-сервер сетевой ФС.

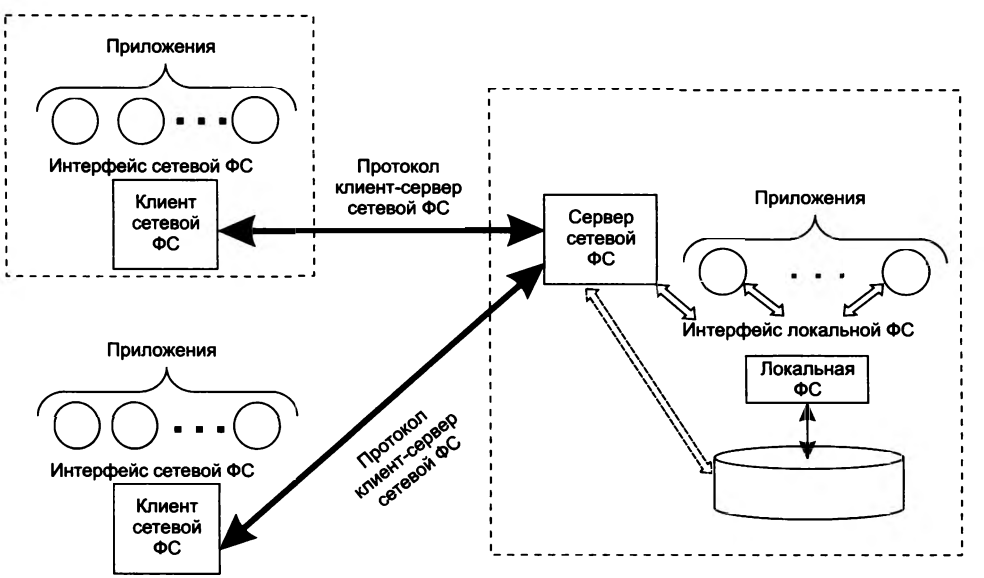


Рис. 24.1. Схема сетевой файловой системы

Клиенты сетевой ФС установлены на многочисленных компьютерах, подключенных к сети. Они обслуживают запросы приложений на доступ к файлам, хранящимся на удаленном

компьютере. В качестве таких приложений часто выступают графические или символьные оболочки ОС, такие как Проводник Windows (Windows Explorer) или Unix shell, а также любые другие пользовательские программы. Клиент сетевой ФС передает по сети запросы серверу сетевой ФС, работающему на удаленном компьютере. Сервер, получив запрос, может обслужить его либо самостоятельно, либо, что является более распространенным вариантом, передать запрос для обслуживания локальной файловой системе. После получения ответа от локальной файловой системы сервер передает его по сети клиенту, а тот, в свою очередь, — приложению, обратившемуся с запросом.

Приложения обращаются к клиенту сетевой ФС, используя определенный программный интерфейс, который в данном случае является *интерфейсом сетевой ФС*. Этот интерфейс стараются сделать как можно более похожим на *интерфейс локальной ФС*, чтобы соблюсти принцип прозрачности. В идеале сетевая файловая система должна выглядеть для пользователя так же, как его локальная файловая система. При полном совпадении интерфейсов приложение может обращаться к локальным и удаленным файлам и каталогам с помощью одних и тех же системных вызовов, совершенно не принимая во внимание места хранения данных. Такой эффект достигается, если, например, при использовании на сервере сети локальной файловой системы FAT интерфейс сетевой файловой системы повторяет системные вызовы FAT.

Клиент и сервер взаимодействуют друг с другом через сеть по *протоколу сетевой файловой системы*. Если интерфейсы локальной и сетевой файловых систем совпадают, этот протокол может быть достаточно простым — в его функции должна входить ретрансляция серверу запросов, принятых клиентом от приложений, с которыми тот затем будет обращаться к локальной ФС. Протокол сетевой ФС, помимо простой ретрансляции системных файловых вызовов от клиента серверу, может выполнять и более сложные функции, учитывающие природу сетевого взаимодействия, например то, что клиент и сервер работают на разных компьютерах, которые могут оказаться неработоспособными, или что сообщения передаются по ненадежной и вносящей порой большие задержки сетевой среде.

В качестве иллюстрации приведенной модели давайте рассмотрим сетевую файловую систему, построенную на базе локальной файловой системы FAT и использующую в качестве протокола клиент-сервер протокол **SMB** (Server Message Block). Расширенные версии протокола SMB получили название CIFS (Common Internet File System). Протокол SMB/CIFS является основой сетевой файловой службы в операционных системах семейства Windows компании Microsoft. Работа протокола начинается с того, что клиент отправляет серверу специальное сообщение с запросом на установление соединения. В процессе установления соединения SMB-клиент и SMB-сервер обмениваются информацией о себе: они сообщают друг другу, какой диалект протокола SMB они будут использовать в этом соединении (диалект здесь — определенное подмножество функций протокола, так как помимо файловых функций SMB поддерживает доступ к принтерам, управление внешними устройствами и некоторые другие). Если сервер готов к установлению соединения, он отвечает сообщением-подтверждением. После установления соединения клиент может обращаться к серверу, передавая ему в SMB-сообщениях команды манипулирования файлами и каталогами. Клиент может запросить сервер создать и удалить каталог, предоставить содержимое каталога, создать и удалить файл, прочитать и записать содержимое файла, установить атрибуты файла и т. п.

В среде операционной системы Unix наибольшее распространение получили две сетевые файловые системы и соответственно два протокола клиент-сервер — **FTP** (File Transfer Protocol) и **NFS** (Network File System).

Факторы эффективности ФС

Основными показателями эффективности файловой системы являются ее производительность, надежность, простота обслуживания и удобство работы пользователя.

Производительность сетевой файловой системы характеризуется скоростью доступа к файлам и временем выполнения требуемых операций с ними.

На производительность ФС влияют различные факторы, в том числе:

- число пользователей, которые работают в системе: чем их больше, тем вероятнее возникновение больших задержек обслуживания из-за заторов в сети и перегрузки файлового сервера;
- единица данных, перемещаемых между сервером и клиентом (часть файла или файл целиком);
- архитектурные решения, определяющие, например, режим работы компонентов ФС (режим ядра или пользовательский режим);
- способ *кэширования* передаваемых данных, и в частности, используется кэширование на стороне клиента или на стороне сервера.

На *надежность* файловой системы прямое влияние оказывают отказы оборудования и программного обеспечения. При этом последствия отказов зависят от функциональной роли отказавшего компонента — был ли это клиентский компьютер, сервер или произошел разрыв связи между ними.

Отказы клиентских компьютеров, на которых работают клиентские программы, ведут к деградации файловой службы — отказавший компьютер исключается из зоны предоставления услуг.

Последствия, к которым может привести разрыв связи, зависят от топологии сети и локализации повреждения. Степень деградации может варьироваться в широких пределах. Это может быть один или несколько «потерянных» клиентов в случае нарушения линий связи, присоединяющих группу клиентских компьютеров. Однако возможен и полный отказ системы, если при отсутствии резервирования произошел разрыв связи файлового сервера с остальной частью сети.

К аналогичным последствиям приводят отказы компьютеров, выступающих в роли файловых серверов.

Существует ряд способов смягчения последствий отказа сервера и разрыва связи, они базируются на концепциях *запоминания* (stateful) и *незапоминания* (stateless) *состояния* режимов работы клиентов и серверов.

Некоторые технологии, используемые в файловых системах, положительно влияют как на производительность, так и на надежность, к таким технологиям относится *репликация* — поддержание нескольких копий одного и того же файла. Так же как и при кэшировании, здесь возникает ситуация, когда одни и те же данные могут независимо модифицироваться разными пользователями. Для этих целей в сетевой ФС должна быть определена *семантика разделения файлов*.

Модели загрузки-выгрузки и удаленного доступа

В зависимости от принятой в ней *единицы перемещения данных* сетевая файловая система может быть отнесена к одной из двух моделей:

- загрузки-выгрузки;
- удаленного доступа.

В **модели загрузки-выгрузки** процедура передачи данных включает следующие шаги: чтение файла *целиком* с сервера на машину клиента, обработка файла на машине клиента, запись обновленного файла на сервер. Типичным представителем этого вида файлового интерфейса является служба FTP, пользователь которой должен применить команду `get file_name` для перемещения файла с сервера на клиентский компьютер и команду `put file_name` для возвращения файла на сервер. Таким образом, сетевой интерфейс ФС такого типа включает в себя только команды хранения и перемещения файла, а все операции с файлом выполняются на основе интерфейса локальной файловой системы.

Другой тип сетевой файловой системы соответствует **модели удаленного доступа**, которая предполагает поддержку большого количества операций с удаленными файлами: открытие и закрытие файлов, чтение и запись *частей* файла, позиционирование в файле, проверка и изменение атрибутов файла и т. д. В отличие от модели загрузки-выгрузки, все файловые операции в данном случае выполняются на серверах, а клиенты только генерируют запросы на их отработку. Преимуществом такого подхода являются низкие требования к дисковому пространству на клиентских машинах, а также исключение необходимости передачи целого файла, когда нужна только его часть. Модель удаленного доступа часто используется в прозрачных реализациях файлового интерфейса, когда удаленные файловые системы монтируются в общее с локальной системой дерево (или его часть, что происходит при отображении удаленной системы на букву логического диска). Модели удаленного доступа могут использовать различные наименьшие единицы перемещения части файла. Наиболее популярны такие единицы, как байт, блок или запись. Последний вид единицы может применяться только в том случае, когда локальная файловая система поддерживает структурированные файлы.

При применении модели удаленного доступа для повышения производительности может выполняться также *буферизация команд*, задающих операции с файлом. Команды группируются и передаются для выполнения на сервер в виде одного пакета, тем самым снижается сетевой трафик, а значит, повышается производительность сети.

Файловые серверы с запоминанием и без запоминания состояния

Отказ файлового сервера во время сеанса связи с клиентом может приводить к разным последствиям в зависимости от того, *запоминает этот сервер состояние* (stateful) или *не запоминает* (stateless).

Режим запоминания состояния применяется не только к сетевым, но и к локальным ФС. Для пояснения рассмотрим работу приложения с файлом. Чаще всего с одним и тем же

файлом приложение выполняет не одну, а несколько операций. Какие бы операции над файлом ни выполнялись — `read` или `write`, `creat` или `delete`, — они непременно включают ряд *универсальных* действий, повторяемых в неизменном виде при выполнении любой операции. Например, все операции над файлом начинаются с того, что по символьному имени файла определяется местонахождение на диске характеристик файла, описывающих его физическую и логическую структуры, ограничения доступа и др., а затем эти данные переносятся с диска в оперативную память.

Помимо универсальных действий, каждая операция с файлом включает ряд *специфических* для этой операции действий.

Пусть имеется некая последовательность операций с одним и тем же файлом `X`, например: `read(X)`, `write(X)`... Операционная система может выполнить эту последовательность двумя способами.

- Первый подход состоит в последовательном выполнении для каждой операции всех относящихся к ней действий, как универсальных, так и специфических. То есть когда из прикладной программы поступает системный вызов на выполнение операции `read(X)`, ФС ее выполняет, отсылает ответ, а затем удаляет из своих внутренних таблиц в оперативной памяти всю информацию о выполненной операции и «с чистого листа» приступает к выполнению операции `write(X)`.
- В соответствии со вторым подходом универсальные действия выполняются лишь в начале и в конце последовательности операций, а для каждой промежуточной операции выполняются только специфические действия. То есть если первой была выполнена операция чтения `read(X)`, то при выполнении следующей операции `write(X)` вовсе не требуется снова производить поиск и перенос в оперативную память адреса файла и других его характеристик. Действительно, эта информация уже была найдена во время выполнения предыдущей операции `read(X)`, достаточно просто ее сохранить для повторного использования.

В первом случае файловую систему относят к типу **ФС без запоминания состояния операций** (stateless), а во втором — к **ФС с запоминанием состояния операций** (stateful).

Преимуществом первого способа является его большая устойчивость к сбоям в работе системы, так как каждая операция является самодостаточной и не зависит от результата предыдущей.

Однако большинство файловых систем поддерживает второй способ организации, когда файловая система сохраняет информацию о предыдущих операциях, поскольку этот способ более экономичный и быстрый.

В сетевых ФС могут быть реализованы две схемы. Одна из них предполагает хранение информации о состоянии текущих операций на сервере, другая — на клиентском компьютере. То есть файловый сервер во втором варианте не запоминает состояние, а клиент запоминает.

Если произойдет крах системы, работающей по первой схеме, то после перезагрузки сервера информация о текущих операциях потеряется, так что приложение, работающее на клиентском компьютере, не сможет продолжить нормальную работу с открытыми до краха файлами.

Вторая схема позволяет приложениям выйти из такой ситуации с меньшими потерями. Перезагрузка сервера приводит только к паузе в обслуживании, но работа с файлами может быть после этого продолжена безболезненно для клиента. Однако при такой организации

каждый запрос от stateful-клиента к stateless-серверу должен содержать исчерпывающую информацию (полное имя файла, смещение в файле и т. п.), необходимую серверу для выполнения требуемой операции. Очевидно, что эта информация увеличивает длину сообщения и время, которое тратит сервер на локальное открытие файла каждый раз, когда над ним производится очередная операция чтения или записи.

Семантика разделения файлов

Организация сетевой файловой системы во многом определяется принятой в этой системе *семантикой разделения файлов*. Когда два или более пользователя совместно работают с одним и тем же файлом, необходимо совершенно точно определить, что понимается под операциями чтения, записи и сохранения файла, иначе могут возникнуть проблемы с интерпретацией результирующих данных файла. Например, если два пользователя одновременно работают с файлом и один из пользователей сделал несколько записей в файл, то должны ли они немедленно появиться на экране компьютера другого пользователя? Или они должны накапливаться и вноситься в файл только во время его закрытия? А не будут ли при этом потеряны изменения, сделанные в тот же период другим пользователем? Такого рода вопросы допускают несколько вариантов ответа, соответствующих разным семантикам.

- *Семантика Unix.* В централизованных многопользовательских операционных системах, разрешающих разделение файлов, таких как Unix (имеется в виду локальная версия этой ОС), обычно полагается, что когда операция чтения следует за операцией записи, читается уже обновленный файл. Соответственно когда операция чтения следует за двумя операциями записи, то читается файл, измененный последней операцией записи. Тем самым система придерживается абсолютного временно́го упорядочивания всех операций и всегда возвращает самое последнее значение данных. Если запись осуществляется в файл, открытый несколькими пользователями, то все пользователи немедленно видят результат изменения данных файла. Обычно такая модель называется семантикой Unix. В централизованной системе, где файлы хранятся в единственном экземпляре, ее легко и понять, и реализовать.

  Семантика Unix может поддерживаться и в распределенных системах, но только если в ней имеется лишь один файловый сервер и клиенты не кэшируют файлы. Для этого все операции чтения и записи направляются на файловый сервер, который обрабатывает их строго последовательно. На практике, однако, производительность распределенной системы, в которой все запросы к файлам идут на один сервер, часто оказывается неудовлетворительной. Эта проблема иногда решается за счет разрешения клиентам обрабатывать локальные копии часто используемых файлов в своих личных кэшах. Если клиент сделает локальную копию файла в своем локальном кэше и начнет ее модифицировать, а вскоре после этого другой клиент прочитает этот файл с сервера, то он получит неверную копию файла. Одним из способов устранения этого недостатка является немедленный возврат всех изменений в кэшированном файле на сервер. Такой подход хотя и концептуально прост, но не слишком эффективен. Распределенные файловые системы обычно используют более свободную семантику разделения файлов.

- *Сеансовая семантика.* В соответствии с этой моделью изменения в открытом файле сначала видны только процессу, который модифицирует файл, и только после закрытия файла

эти изменения могут видеть другие процессы. В случае сеансовой семантики возникает проблема одновременного использования одного и того же файла двумя или более клиентами. Одним из решений этой проблемы является принятие правила, в соответствии с которым окончательным является тот вариант файла, который был закрыт последним. Однако из-за задержек в сети часто оказывается трудным определить, какая из копий файла была закрыта последней. Менее эффективным, но гораздо более простым в реализации является вариант, в котором окончательным результирующим файлом на сервере считается любой из этих файлов, то есть результат операций с файлом недетерминирован.

- *Семантика неизменяемых файлов.* Следующий подход к разделению файлов заключается в том, чтобы сделать все файлы неизменяемыми. Тогда файл нельзя открыть для записи, а можно выполнять только операции `create` (создать) и `read` (читать). В результате для изменения файла остается единственная возможность — создать полностью новый файл и поместить его в каталог под новым именем. Следовательно, хотя файл и нельзя модифицировать, его можно заменить (автоматически) новым файлом. Для неизменяемых файлов намного проще осуществлять кэширование и репликацию (тиражирование), так как исключаются проблемы, связанные с обновлением всех копий файла при его изменении.

  Таким образом, для файловых систем, работающих с немодифицируемыми файлами, исключаются все проблемы, связанные с одновременным доступом к файлам нескольких пользователей. Однако решение этих проблем перекладывается на плечи пользователей, которые в такой системе вынуждены вести учет многочисленных версий файла, возникающих при его модификации. Пользователи должны поддерживать систему именования файлов, отражающую тот факт, что все множество порожденных файлов имеет близкое содержание, и в то же время позволяющую различать версии файлов.

Кэширование

Кэширование данных в оперативной памяти может существенно повысить скорость доступа к файлам, хранящимся на дисках, независимо от того, является файловая система локальной или сетевой. В сетевых файловых системах кэширование позволяет не только повысить скорость доступа к удаленным данным (это по-прежнему является основной целью кэширования), но и улучшить масштабируемость и повысить надежность файловой системы.

Схемы кэширования, применяемые в сетевых файловых системах, отличаются решениями по трем ключевым вопросам:

- месту расположения кэша;
- способу распространения модификаций;
- проверке достоверности кэша.

Место расположения кэша

В системах, состоящих из клиентов и серверов, потенциально имеется три различных места для хранения кэшируемых файлов и их частей: память сервера, диск клиента (если имеется) и память клиента.

Память сервера практически всегда применяется для кэширования файлов, к которым обращаются по сети пользователи и приложения. *Кэширование в памяти сервера* сокращает время доступа по сети за счет исключения времени обмена с диском сервера. При этом файловый сервер может использовать для кэширования своих файлов существующий механизм локального кэша операционной системы, не применяя никакого дополнительного программного кода. Однако кэширование только в памяти сервера не решает всех проблем — скорость доступа по-прежнему снижают задержки, вносимые сетью, а также перегруженность процессора сервера при одновременном обслуживании большого потока сетевых запросов от многочисленных клиентов.

Кэширование на стороне клиента, которое подразделяется на кэширование на диске клиента и в памяти клиента, исключает обмен по сети и освобождает сервер от работы после переноса файла на клиентский компьютер. Использование диска как места временного хранения данных на стороне клиента позволяет кэшировать большие файлы, что особенно важно при применении модели загрузки-выгрузки, в соответствии с которой файлы переносятся целиком. Диск также является более надежным устройством хранения информации по сравнению с оперативной памятью. Однако такой способ кэширования вносит дополнительные задержки доступа, связанные с чтением данных с клиентского диска, кроме того, он неприменим на бездисковых компьютерах.

Кэширование в оперативной памяти клиента позволяет ускорить доступ к данным, но ограничивает размер кэшируемых данных объемом памяти, выделяемой под кэш, что может стать существенным ограничением при применении модели загрузки-выгрузки.

Распространение модификаций

Существование в одно и то же время в сети нескольких копий одного и того же файла, хранящихся в кэшах клиентов, порождает проблему согласования копий, которая состоит в том, что модификации данных одной из копий должны быть своевременно распространены на все остальные копии.

Существует несколько вариантов распространения модификаций. От выбранного варианта в значительной степени зависит семантика разделения файлов.

Одним из путей решения проблемы согласования является использование **алгоритма сквозной записи**. Когда кэшируемый элемент (файл или блок) модифицируется, новое значение записывается в кэш и одновременно посылается на сервер для обновления главной копии файла. В результате другой процесс, читающий этот файл, получает самую последнюю версию данных. Данный вариант распространения модификаций обеспечивает семантику разделения файлов в стиле Unix.

Один из недостатков алгоритма сквозной записи состоит в том, что он уменьшает интенсивность сетевого обмена только при чтении, при записи интенсивность сетевого обмена та же самая, что и без кэширования. Многие разработчики систем находят это неприемлемым и предлагают алгоритм отложенной записи, основанный на буферизации изменений: вместо того чтобы выполнять запись на сервер, клиент просто делает пометку, что файл изменен. Примерно каждые 30 секунд все изменения в файлах собираются вместе и отсылаются на сервер за один прием. Одна длинная запись для сетевого обмена обычно более эффективна, чем множество коротких.

Следующим шагом в этом направлении является принятие сеансовой семантики, в соответствии с которой запись файла на сервер производится только после закрытия файла. Этот алгоритм, называемый **записью по закрытию**, приводит к тому, что если две копии одного файла кэшируются на разных машинах и последовательно записываются на сервер, то второй записывается поверх первого. Данная схема не может снизить сетевой трафик, если объем изменений за сеанс редактирования файла невелик.

Для всех схем, связанных с задержкой записи, характерна низкая надежность, так как все модификации, не отправленные на сервер на момент краха системы, теряются. Кроме того, задержка делает семантику совместного использования файлов не очень ясной, поскольку данные, которые считывает процесс из файла, зависят от соотношения момента чтения с моментом очередной записи модификаций.

Проверка достоверности кэша

Распространение модификаций решает только проблему согласования с клиентскими копиями главной копии файла, хранящейся на сервере. В то же время этот прием не дает никакой информации о том, когда должны обновляться данные, находящиеся в кэшах клиентов. Очевидно, что данные в кэше одного клиента становятся недостоверными, когда данные, модифицированные другим клиентом, переносятся в главную копию файла. Следовательно, необходимо каким-то образом проверять, являются ли данные в кэше клиента достоверными. В противном случае данные кэша должны быть повторно считаны с сервера. Существует два подхода к решению этой проблемы: инициирование проверки клиентом и инициирование проверки сервером.

В первом случае клиент связывается с сервером и проверяет, соответствуют ли данные в его кэше данным главной копии файла на сервере. Клиент может выполнять такую проверку одним из трех способов.

- *Проверка перед каждым доступом к файлу* дискредитирует саму идею кэширования, так как каждое обращение к файлу вызывает обмен по сети с сервером, но обеспечивает семантику разделения файлов Unix.
- *Периодические проверки* повышают производительность, но делают семантику разделения неясной, зависящей от временны́х соотношений.
- *Проверка при открытии файла* подходит для сеансовой семантики. Отметим, что сеансовая семантика требует одновременного выполнения трех условий: во-первых, файловая система должна соответствовать модели загрузки-выгрузки; во-вторых, модификации должны распространяться в соответствии с алгоритмом записи по закрытию; в-третьих, проверка достоверности кэша должна происходить при открытии файла.

Совершенно другой подход к проблеме проверки достоверности кэша реализуется при инициировании проверки сервером, который также можно назвать методом централизованного управления. Когда файл открывается, то клиент, выполняющий это действие, посылает соответствующее сообщение файловому серверу, в котором указывает режим открытия файла — чтение или запись. Файловый сервер сохраняет информацию о том, кто и какой файл открыл, а также о том, открыт файл для чтения, для записи или для того и другого. Если файл открыт для чтения, то нет никаких препятствий для разрешения другим процессам открыть его для чтения, но открытие его для записи должно быть запрещено. Аналогично, если некоторый процесс открыл файл для записи, то все другие виды доступа

должны быть запрещены. При закрытии файла также необходимо оповестить файловый сервер для того, чтобы он обновил свои таблицы, содержащие данные об открытых файлах. Модифицированный файл также может быть выгружен на сервер в такой момент.

Когда новый клиент делает запрос на открытие уже открытого файла и сервер обнаруживает, что режим нового открытия входит в противоречие с режимом текущего открытия, то сервер может ответить на такой запрос следующими способами:

- отвергнуть запрос;
- поместить запрос в очередь;
- запретить кэширование для данного конкретного файла, потребовав от всех клиентов, открывших этот файл, удалить его из кэша.

Подход, основанный на централизованном управлении, весьма эффективен, обеспечивает семантику разделения Unix, но обладает несколькими недостатками.

- Он отклоняется от традиционной модели взаимодействия клиента и сервера, в которой сервер только отвечает на запросы, инициированные клиентами. Это делает код сервера нестандартным и достаточно сложным.
- Сервер обязательно должен хранить информацию о состоянии сеансов клиентов, то есть работать по схеме с запоминанием состояния (stateful).
- По-прежнему должен использоваться механизм инициирования проверки достоверности клиентами при открытии файлов.

Репликация

Чтобы избежать потери данных и разрушения целостности файловой системы при сбоях и отказах серверов, а также при разрыве критически важных линий связи, прибегают к **резервированию**. Частным случаем резервирования является репликация.

> **Репликация (replication) — это метод поддержания нескольких копий одного и того же файла, каждая из которых хранится на отдельном файловом сервере, при этом обеспечивается автоматическое согласование данных в копиях файла.**

Для каждого файла (или целиком для локальной файловой системы) в сети поддерживается по крайней мере две копии. Протокол сетевого доступа к файлам должен учитывать такую организацию файловой службы: например, в случае отказа одного файлового сервера переадресовывать запрос к другому серверу, работоспособному и поддерживающему реплику требуемого файла.

Репликация файлов не только повышает отказоустойчивость, но и решает проблему перегрузки файловых серверов, так как запросы к файлам распределяются между несколькими серверами и повышают производительность сетевой файловой системы.

Репликация похожа на кэширование файлов на стороне клиентов тем, что в системе создается несколько копий одного файла. Однако существуют и принципиальные отличия репликации от кэширования, прежде всего в преследуемых целях. Если кэширование предназначено для обеспечения локального доступа к файлу одному клиенту и повышения за счет этого скорости работы этого клиента, то репликация нужна для повышения

надежности хранения файлов и снижения нагрузки на файловые серверы. Реплики файла доступны всем клиентам, так как хранятся на файловых серверах, а не на клиентских компьютерах, и о существовании реплик известно всем компьютерам сети. Файловая система обеспечивает достоверность данных реплики и защиту ее данных.

Прозрачность репликации

Ключевым вопросом, связанным с репликацией, является прозрачность. До какой степени пользователи должны быть в курсе того, что некоторые файлы реплицируются? Должны они играть какую-либо роль в процессе репликации или репликация должна выполняться полностью автоматически? В одних системах пользователи полностью вовлечены в этот процесс, в других система все делается без их ведома. В последнем случае говорят, что система репликационно прозрачна.

Прозрачность репликации зависит от двух факторов: выбранной схемы именования реплик и степени вовлеченности пользователя в управление репликацией.

Именование реплик. Прозрачность доступа к файлу, существующему в виде нескольких реплик, может поддерживаться системой именования, которая отображает имя файла на его сетевой идентификатор, однозначно определяющий место хранения файла. Если в сети используется справочная служба, которая позволяет хранить отображения имен объектов на их сетевые идентификаторы (например, IP-адреса серверов), то для реплицированных файлов допустима прозрачная схема именования. В этом случае файлу присваивается имя, не содержащее старшей части, соответствующей имени компьютера. В справочной службе этому имени соответствует несколько идентификаторов, указывающих на серверы, хранящие реплики файла. При обращении к такому файлу приложение использует имя, а справочная служба возвращает ему один из идентификаторов, указывающий на сервер, хранящий реплику.

Наиболее просто такую схему реализовать для неизменяемых файлов, все реплики которых всегда (или почти всегда, если файлы редко, но все же изменяются) идентичны. В случае, когда реплицируются изменяемые файлы, полностью прозрачный доступ требует ведения некоторой базы, хранящей сведения о том, какие из реплик содержат последнюю версию данных, а какие еще не обновлены. Для поддержания полностью прозрачной системы репликации необходимо также постоянное использование в файловом интерфейсе имен, не зависящих от местоположения, то есть не содержащих старшей части с указанием имени сервера. В более распространенной на сегодня схеме именования требуется явное указание имени сервера при обращении к файлу, что приводит к не полностью прозрачной системе репликации.

Управление репликацией подразумевает определение количества реплик и выбор серверов для хранения каждой реплики. В прозрачной системе репликации такие решения принимаются автоматически при создании файла на основе правил стратегии репликации, определенных заранее администратором системы. В непрозрачной системе репликации решения о количестве реплик и их размещении принимаются с участием пользователя, который создает файлы, или разработчика приложения, если файлы создаются в автоматическом режиме. В результате существует два режима управления репликацией.

- При *явной репликации* (explicit replication) пользователю (или прикладному программисту) предоставляется возможность управления процессом репликации. При созда-

нии файла сначала создается первая реплика с явным указанием сервера, на котором размещается файл, а затем создается несколько реплик, причем для каждой реплики сервер также указывается явно. Пользователь при желании может впоследствии удалить одну или несколько реплик. Явная репликация не исключает автоматического режима поддержания согласованности реплик, которые создал и разместил на серверах пользователь.

- При *неявной репликации* (implicit replication) выбор количества и места размещения реплик производится в автоматическом режиме без участия пользователя. Приложение не должно указывать место размещения файла при запросе на его создание. Файловая система самостоятельно выбирает сервер, на который помещает первую реплику файла. Затем в фоновом режиме система создает несколько реплик этого файла, выбирая их количество и серверы для их размещения. Если надобность в некоторых репликах исчезает (это также определяется автоматически), то система их удаляет.

Согласование реплик

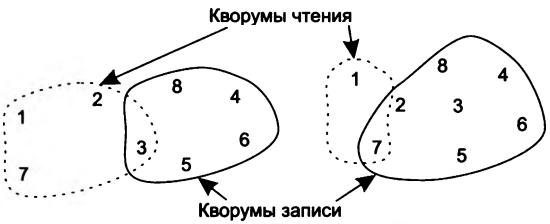
Согласование реплик, в результате чего в каждой реплике должна сохраняться последняя версия данных файла, — одна из наиболее важных проблем при разработке системы репликации. Так как изменяемые файлы являются самым распространенным типом файлов, то в какой-то момент времени данные в одной из реплик модифицируются, после чего требуется предпринять усилия для распространения модификаций на все остальные реплики. Существует несколько способов обеспечения согласованности реплик.

- *Чтение любого — запись во все* (Read-Any — Write-All). При необходимости записи в файл все реплики файла блокируются, затем выполняется запись в каждую копию, после чего блокировка снимается и файл становится доступным для чтения. Чтение может выполняться из любой копии. Этот способ обеспечивает семантику разделения файлов в стиле Unix. Недостатком является то, что запись не всегда можно осуществить, так как некоторые серверы, хранящие реплики файла, могут в момент записи быть неработоспособными.
- *Запись в доступные* (Available-Copies). Этот метод снимает ограничение предыдущего, так как запись выполняется только в те копии, серверы которых доступны на момент записи. Чтение осуществляется из любой реплики файла, как и в предыдущем методе. Любой сервер, хранящий реплику файла, после перезагрузки должен соединиться с другим сервером и получить от него обновленную копию файла и только потом начать обслуживать запросы на чтение из файла. Для обнаружения отказавших серверов в системе должен работать специальный процесс, постоянно опрашивающий состояние серверов. Недостатком метода является возможность появления несогласованных копий файла из-за коммуникационных проблем в сети, когда невозможно выявить отказавший сервер.
- *Первичная реплика* (Primary-Copy). В этом методе запись разрешается только в одну реплику файла, называемую первичной (primary). Все остальные реплики файла являются вторичными (secondary), из них можно только читать данные. После модификации первичной реплики все серверы, хранящие вторичные реплики, должны связаться с сервером, поддерживающим первичную реплику, и получить от него обновления. Этот процесс может инициироваться как первичным сервером, так и вторичными

(периодически проверяющими состояние первичной реплики). Этот метод является одной из реализаций метода «чтение любой — запись во все», в которой процесс записи реализован иерархическим способом. Для аккумулирования нескольких модификаций и сокращения сетевого трафика распространение модификаций может быть выполнено с запаздыванием, но в этом случае возникает проблема согласованности копий. Недостатком метода является его низкая надежность — при отказе первичного сервера модификации файла становятся невозможными (для решения этой проблемы необходимо повысить статус некоторого вторичного сервера до первичного).

- *Кворум* (Quorum). Этот метод обобщает подходы, реализованные в предыдущих методах. Пусть в сети существует n реплик некоторого файла. Алгоритм основан на том, что при модификации файла изменения записываются в w реплик, а при чтении файла клиент обязательно производит обращение к r репликам. Значения w и r выбираются так, что $w + r > n$. При чтении клиент должен иметь возможность сначала проверить версию каждой реплики, а затем выбрать старшую и читать данные уже из нее. Очевидно, что при модификации файла номер версии должен наращиваться, а если при записи в w реплик они имеют разные версии, то выбирается максимальное значение версии для наращивания и присваивания всем репликам.

  При выполнении условия $w + r > n$ среди любых выбранных произвольным образом r реплик всегда найдется хотя бы одна из w реплик, в которую были записаны последние обновления. Так, если реплик восемь ($n = 8$), можно выбрать в качестве r значение 4, а в качестве w — значение 5 (рис. 24.2). Условие $w + r > n$ при этом удовлетворяется.



а) $n = 8$; $r = 4$; $w = 5$ б) $n = 8$; $r = 2$; $w = 7$

Рис. 24.2. Примеры работы метода кворума

Предположим, что запись последних изменений была выполнена в реплики с номерами 3, 4, 5, 6, 8. Если при чтении выбраны реплики 1, 2, 3, 7, то реплика 3 окажется общей для операций записи и чтения, поэтому прочитаны будут корректные данные (рис. 24.2, *а*). В другом примере, который иллюстрирует рис. 24.2, *б*, значение r выбрано равным 2, а w — 7. Результат также получен корректный.

У метода кворума имеются частные случаи. Так, если $r = 1$, а $w = n$, то получаем метод «чтение любого — запись во все».

Еще одним немаловажным параметром процесса согласования копий является величина временной задержки между внесением изменений в одной из реплик и отображением этих модификаций на все остальные реплики. Существует два подхода к распространению модификаций: **строгая целостность** (tight consistency) и **нестрогая целостность** (loose consistency). В первом случае идентичность реплик гарантируется в любой момент времени. Такой результат достигается, например, при использовании механизма неделимых

транзакций. При этом сеть должна обладать высокими показателями производительности и надежности, чтобы обеспечить постоянную доступность всех узлов. Во втором случае между внесением изменений и их воспроизведением в остальных репликах допускается наличие временно́й задержки. Этот вариант более приемлем для современных файловых систем. Обычно допустимые временны́е задержки здесь измеряются значениями от нескольких секунд до нескольких минут.

Сетевая файловая служба на основе протокола FTP

Сетевая ФС на основе протокола **FTP** (File Transfer Protocol) представляет собой одну из наиболее ранних служб доступа к удаленным файлам. До появления службы WWW это была самая популярная служба доступа к удаленным данным в Интернете и корпоративных IP-сетях. Первые спецификации FTP относятся к 1971 году. FTP-серверы и FTP-клиенты имеются практически в каждой ОС семейства Unix, а также во многих других сетевых ОС. FTP-клиенты встроены сегодня в программы просмотра (браузеры) Интернета, так как архивы файлов на основе протокола FTP по-прежнему популярны, и для доступа к таким архивам браузер использует протокол FTP.

Протокол FTP целиком перемещает файл с удаленного компьютера на локальный и наоборот, то есть работает по схеме загрузки-выгрузки. Кроме того, он поддерживает несколько команд просмотра удаленного каталога и перемещения по каталогам удаленной файловой системы. В протокол FTP встроены примитивные средства аутентификации удаленных пользователей на основе передачи по сети пароля в открытом виде. Кроме того, поддерживается анонимный доступ, не требующий указания имени пользователя и пароля. Анонимный доступ является более безопасным, так как не подвергает пароли пользователей угрозе перехвата.

Протокол FTP выполнен по схеме клиент-сервер. FTP-клиент включает три функциональных модуля.

- **User Interface** — модуль, поддерживающий интерфейс клиента с пользователем. Он принимает от пользователя символьные команды и воспроизводит состояние FTP-сеанса на символьном экране. Наряду с традиционными клиентами, работающими в символьном режиме, имеются и графические оболочки, не требующие от пользователя знания символьных команд. Символьные клиенты обычно поддерживают следующий основной набор команд:
  - `open` *имя_хоста* — открытие сеанса с удаленным сервером;
  - `bye` — завершение сеанса с удаленным хостом и завершение работы утилиты ftp;
  - `close` — завершение сеанса с удаленным хостом, утилита ftp продолжает работать;
  - `ls (dir)` — печать содержимого текущего удаленного каталога;
  - `get` *имя_файла* — копирование удаленного файла на локальный хост;
  - `put` *имя_файла* — копирование локального файла на удаленный сервер.
- **User-PI** — интерпретатор команд пользователя. Этот модуль взаимодействует с соответствующим модулем FTP-сервера.

- **User-DTP** — модуль, осуществляющий передачу данных файла по командам, получаемым от модуля User-PI по протоколу клиент-сервер. Этот модуль взаимодействует с локальной файловой системой клиента.

FTP-сервер включает два модуля.

- **Server-PI** — модуль, который принимает и интерпретирует команды, передаваемые по сети модулем User-PI.
- **Server-DTP** — модуль, управляющий передачей данных файла по командам от модуля Server-PI. Взаимодействует с локальной файловой системой сервера.

FTP-клиент и FTP-сервер поддерживают параллельно два сеанса — управляющий сеанс и сеанс передачи данных. Управляющий сеанс открывается при установлении первоначального FTP-соединения клиента с сервером, причем в течение одного управляющего сеанса может последовательно проходить несколько сеансов передачи данных, в рамках которых передается или принимается несколько файлов.

Общая схема взаимодействия клиента и сервера выглядит следующим образом.

1. FTP-сервер всегда открывает управляющий TCP-порт 21 для прослушивания, ожидая прихода запроса на установление управляющего FTP-сеанса от удаленного клиента.
2. После установления управляющего соединения клиент отправляет на сервер команды, которые уточняют параметры соединения: имя и пароль клиента, роль участников соединения (активная или пассивная), порт передачи данных, тип передачи, тип передаваемых данных (двоичные данные или ASCII-код), директивы на выполнение действий (читать файл, писать файл, удалить файл и т. п.).
3. После согласования параметров пассивный участник соединения переходит в режим ожидания открытия соединения на порт передачи данных. Активный участник инициирует открытие соединения и начинает передачу данных.
4. После окончания передачи данных соединение по портам данных закрывается, а управляющее соединение остается открытым. Пользователь может по управляющему соединению активизировать новый сеанс передачи данных.

Порты передачи данных выбирает FTP-клиент (по умолчанию клиент может задействовать для передачи данных порт управляющего сеанса), а сервер должен использовать порт, на единицу меньший порта клиента.

В протоколе FTP предусмотрены специальные команды для взаимодействия FTP-клиента с FTP-сервером (не следует их путать с командами пользовательского интерфейса клиента, ориентированные на применение человеком). Эти команды делятся на три группы.

- *Команды управления доступом к системе* доставляют серверу имя и пароль клиента, изменяют текущий каталог на сервере, повторно инициализируют, а также завершают управляющий сеанс.
- *Команды управления потоком данных* устанавливают параметры передачи данных. Служба FTP может применяться для передачи разных типов данных (ASCII-код или двоичные данные), работать как со структурированными данными (файл, запись, страница), так и с неструктурированными.
- *Команды службы FTP* управляют передачей файлов, операциями над удаленными файлами и каталогами. Например, команды `RETR` и `STOR` запрашивают передачу файла соответственно от сервера на клиентский хост и наоборот. Параметрами каждой из этих

команд является имя файла. Может быть задано также смещение от начала файла — это позволяет начать передачу файла с определенного места при непредвиденном разрыве соединения. Команды `DELE`, `MKD`, `RMD`, `LIST` соответственно удаляют файл, создают каталог, удаляют каталог и передают список файлов текущего каталога. Каждая команда протокола FTP передается в виде одной строки ASCII-кода.

Протокол FTP не защищен от перехвата данных по сети, в том числе от перехвата пароля пользователя, этот недостаток устранен в его безопасной версии **SFTP**, построенной на основе протокола SSL[1].

Архитектурные решения ФС

Свойства сетевой ФС во многом определяются архитектурными решениями, принятыми при ее реализации.

Именно к вопросам программной реализации относятся решения о том, оформлять клиентские и серверные компоненты сетевой файловой системы в виде приложений или включать их в число модулей ОС, в каком режиме — пользовательском или привилегированном — должны выполняться эти программы, каким образом распределить функции между клиентом и сервером.

В некоторых файловых системах (например, NFS) на всех компьютерах сети работает одно и то же базовое программное обеспечение, включающее как клиентскую, так и серверную часть, так что любой компьютер, который захочет предложить услуги файловой службы, может играть роль сервера. Для этого администратору ОС достаточно объявить имена выбранных каталогов разделяемыми, чтобы другие машины могли иметь к ним доступ.

В других системах файловый сервер — это специализированный компонент серверной ОС, отсутствующий в клиентских компьютерах. По такой схеме работала, например, популярная в 80-х годах сетевая ОС NetWare, которая предоставляла серверу ФС среду, оптимизированную для его работы.

Для повышения эффективности работы файловые сервер и клиент включают в число модулей ядра ОС, работающих в привилегированном режиме. Эффективность при этом повышается за счет прямого доступа ко всем внутренним модулям ОС без выполнения дополнительных операций и смены режима. Так, например, скорость обмена данными существенно растет при наличии прямого доступа к содержимому дискового кэша, поэтому с целью повышения производительности файлового сервера ОС должна быть сконфигурирована для поддержки дискового кэша большого объема.

(S) *Справочная служба*

Выводы

Файловая служба включает программы-серверы и программы-клиенты, взаимодействующие с помощью определенного протокола по сети между собой.

---

[1] См. раздел «Технологии защищенного канала» в главе 29.

Один компьютер может в одно и то же время предоставлять пользователям сети услуги различных файловых служб.

В сетевой файловой службе в общем случае можно выделить следующие основные компоненты: локальную файловую систему, интерфейс локальной файловой системы, сервер сетевой файловой системы, клиент сетевой файловой системы, интерфейс сетевой файловой системы, протокол клиент-сервер сетевой файловой системы.

В сетевых файловых системах используется различная семантика чтения и записи разделяемых данных, позволяющая избежать проблем с интерпретацией результирующих данных файла.

Файловый интерфейс может быть отнесен к одному из двух типов в зависимости от того, поддерживает ли он модель загрузки-выгрузки или же модель удаленного доступа.

Файловый сервер может быть реализован по одной из двух схем: с запоминанием данных о последовательности файловых операций клиента, то есть с запоминанием состояния (statefull), и без запоминания таких данных, то есть без запоминания состояния (stateless).

Кэширование в сетевых файловых системах позволяет повысить скорость доступа к удаленным данным, масштабируемость и надежность файловой системы.

Репликация подразумевает существование нескольких копий одного и того же файла, каждая из которых хранится на отдельном файловом сервере, при этом обеспечивается автоматическое согласование данных в копиях файла.

Существует несколько способов обеспечения согласованности реплик, которые обобщаются в методе кворума.

Контрольные вопросы

1. Какая модель файлового сервера (с запоминанием или без запоминания состояния) обеспечивает более высокую степень устойчивости к отказам сервера?
2. Какие механизмы сетевой файловой системы положительно сказываются на ее производительности? Варианты ответов:

   а) кэширование;

   б) репликация;

   в) работа в сервера в режиме stateless ;

   г) буферизация команд удаленного доступа к файлу.
3. Какой результат видят на экране два пользователя ОС Unix, набирающие текст в одном файле?
4. Сравните используемые в сетевой ФС два метода кэширования — на стороне клиента и на стороне сервера. Приведите достоинства и недостатки каждого метода.

ГЛАВА 25 Служба управления сетью

Функции систем управления сетью

Как и любой сложный технический объект, компьютерная сеть требует выполнения различных действий для поддержания ее в рабочем состоянии, анализа и оптимизации ее производительности, защиты от внутренних и внешних угроз. Среди многообразия средств, привлекаемых для достижения этих целей, важное место занимают службы (системы) управления сетью.

Система управления сетью (Network Management System, **NMS**) — это сложный программно-аппаратные комплекс, который контролирует сетевой трафик и управляет коммуникационным оборудованием крупной компьютерной сети.

Системы управления сетью работают, как правило, в *автоматизированном* режиме, выполняя наиболее простые действия автоматически и оставляя человеку принятие сложных решений на основе подготовленной системой информации.

Система управления сетью предназначена для решения следующих групп задач:

- *Управление конфигурацией сети и именованием* заключается в конфигурировании параметров как отдельных элементов сети, так и сети в целом. Для элементов сети, таких как маршрутизаторы, мультиплексоры и т. п., конфигурирование состоит в назначении сетевых адресов, идентификаторов (имен), географического положения и пр. Для сети в целом управление конфигурацией обычно начинается с построения карты сети, то есть с отображения реальных связей между элементами сети и связей между ними.
- *Обработка ошибок* включает выявление, определение и устранение последствий сбоев и отказов.
- *Анализ производительности и надежности* связан с оценкой на основе накопленной статистической информации таких параметров, как время реакции системы, пропускная способность реального или виртуального канала связи между двумя конечными абонентами сети, интенсивность трафика в отдельных сегментах и каналах сети, а также вероятность искажения данных при их передаче через сеть. Результаты анализа производительности и надежности позволяют контролировать *соглашение об уровне обслуживания* (SLA), заключаемое между пользователем сети и ее администраторами (или компанией, продающей услуги). Без средств анализа производительности и надежности поставщик услуг публичной сети или отдел информационных технологий предприятия не сможет ни проконтролировать, ни тем более обеспечить нужный уровень обслуживания для конечных пользователей сети.

- *Управление безопасностью* подразумевает контроль доступа к ресурсам сети (данным и оборудованию) и сохранение целостности данных при их хранении и передаче через сеть. Базовыми элементами управления безопасностью являются процедуры аутентификации пользователей, назначение и проверка прав доступа к ресурсам сети, распределение и поддержка ключей шифрования, управления полномочиями и т. п. Часто функции этой группы не включаются в системы управления сетями, а либо реализуются в виде специальных продуктов обеспечения безопасности, например сетевых экранов или централизованных систем авторизации[1], либо входят в состав операционных систем и системных приложений.
- *Учет работы сети* включает регистрацию времени использования различных ресурсов сети (устройств, каналов и транспортных служб) и ведение биллинговых операций (плата за ресурсы).

В стандартах систем управления не делается различий между управляемыми объектами, представляющими коммуникационное оборудование (каналы, сегменты локальных сетей, коммутаторы и маршрутизаторы, модемы и мультиплексоры), и объектами, представляющими аппаратное и программное обеспечение компьютеров. Однако на практике деление систем управления по типам управляемых объектов широко распространено.

В тех случаях, когда управляемыми объектами являются компьютеры, а также их системное и прикладное программное обеспечение, то для системы управления часто используют особое название — система управления системой (System Management System, SMS).

SMS обычно автоматически собирает информацию об установленных в сети компьютерах и создает записи в специальной БД об аппаратных и программных ресурсах. SMS может централизованно устанавливать и администрировать приложения, которые запускаются с серверов, а также удаленно измерять наиболее важные параметры компьютера, операционной системы, СУБД (например, коэффициент использования процессора или физической памяти, интенсивность страничных прерываний и др.). SMS позволяет администратору брать на себя удаленное управление компьютером в режиме эмуляции графического интерфейса популярных операционных систем.

Архитектура систем управления сетью

Агент управляемого объекта

Для решения перечисленных задач необходимо иметь возможность управления отдельным устройством (объектом). Обычно каждое устройство, которое требует достаточно сложного конфигурирования, производитель сопровождает автономной программой конфигурирования и управления, работающей в среде специализированной ОС, установленной на этом устройстве. Мы будем называть такой программный компонент **агентом**. Агенты могут встраиваться в управляемое оборудование либо работать на устройстве, подключенном к интерфейсу управления такого устройства. Один агент в общем случае может управлять несколькими однотипными устройствами.

[1] О средствах обеспечения сетевой безопасности читайте в части VII книги.

Агент поддерживает интерфейс с оператором/администратором, который посылает ему запросы и команды на выполнение определенных операций.

Агент может выполнять следующие функции:

- хранить, извлекать и передавать по запросам извне информацию о технических и конфигурационных параметрах устройства, включая модель устройства, число портов, тип портов, тип ОС, связи с другими устройствами и др.;
- выполнять, хранить и передавать по запросу извне измерения (подсчеты) характеристик функционирования устройства, таких как число принятых пакетов, число отброшенных пакетов, степень заполнения буфера, состояние порта (рабочее или нерабочее);
- изменять по командам, полученным извне, конфигурационные параметры.

В описанной схеме агент играет роль *сервера*, к которому обращается *клиент*-администратор с запросами о значениях характеристик или об установлении конфигурационных параметров управляемого устройства.

Для получения требуемых данных об объекте, а также для выдачи на него управляющих воздействий агент должен иметь возможность взаимодействовать с ним. Многообразие типов управляемых объектов не позволяет стандартизовать способ взаимодействия агента с объектом. Эта задача решается разработчиками при встраивании агентов в коммуникационное оборудование или в операционную систему. Агент может снабжаться специальными датчиками для получения информации, например датчиками температуры. Агенты могут отличаться разным уровнем интеллекта: от минимального, достаточного лишь для подсчета проходящих через оборудование кадров и пакетов, до весьма высокого, позволяющего выполнять последовательности управляющих команд в аварийных ситуациях, строить временны́е зависимости, фильтровать аварийные сообщения и т. п.

Двухзвенная и трехзвенная схемы управления

Среди задач, определенных для систем управления сетью, есть сравнительно редкие операции, например конфигурирование того или иного устройства, а есть и такие, которые требуют частого вмешательства системы (анализ производительности каждого из устройств сети, сбор статистики по загрузке устройств). В первом случае используется «ручное» управление, когда администратор со своей консоли передает команды агенту. Понятно, что такой вариант совсем не подходит для глобального мониторинга всех устройств сети.

Рассмотрим вначале вариант ручного *двухзвенного* управления (рис. 25.1). В качестве протокола взаимодействия клиента и сервера может применяться, например, протокол удаленного управления telnet, клиентская часть которого должна быть установлена на компьютере администратора, а серверная — на устройстве. Серверная часть telnet должна также поддерживать интерфейс с агентом, от которого будет поступать информация о состоянии управляемого объекта и значении его характеристик. На клиентской стороне протокол telnet может быть связан с программой поддержки графического пользовательского интерфейса, которая, например, выводит администратору в графическом виде запрашиваемую характеристику. В общем случае администратор может работать с несколькими агентами.

В качестве протокола взаимодействия клиентской и серверной частей нередко используется протокол веб-службы HTTP.

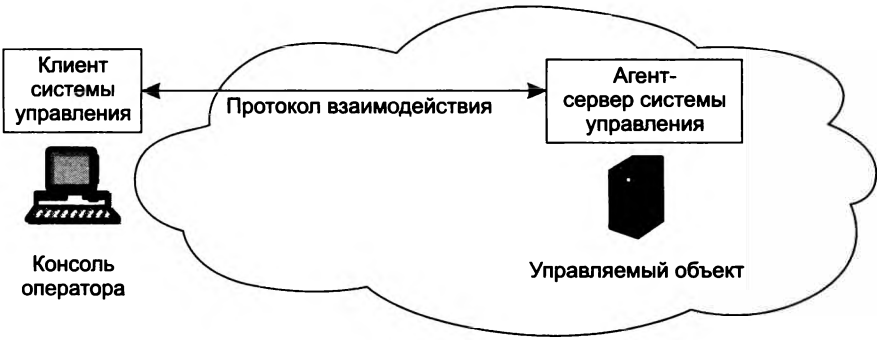


Рис. 25.1. Двухзвенная схема управления устройством

Для задач, требующих частого выполнения операций управления отдельными устройствами, а также при росте количества управляемых устройств рассмотренная схема уже не может решить поставленную задачу. В схему вводится новое промежуточное звено, называемое менеджером. Менеджер призван автоматизировать взаимодействие оператора с множеством агентов. Показанная на рис. 25.2 схема службы управления сетью реализуется в виде *трехзвенного* распределенного приложения, в котором функции между звеньями распределены следующим образом.

- Первое звено — клиент системы управления, устанавливается на компьютере оператора, поддерживает пользовательский интерфейс с промежуточным сервером.
- Второе звено — промежуточный сервер, выполняет функции *менеджера*, устанавливается либо на компьютере оператора, либо на специально выделенном компьютере. Менеджер взаимодействует обычно с несколькими клиентами и агентами, обеспечивая диспетчеризацию запросов клиентов к серверам и обрабатывая полученные от агентов данные в соответствии с поставленными перед системой управления задачами. Для повышения надежности и производительности в системе управления может быть предусмотрено несколько менеджеров.

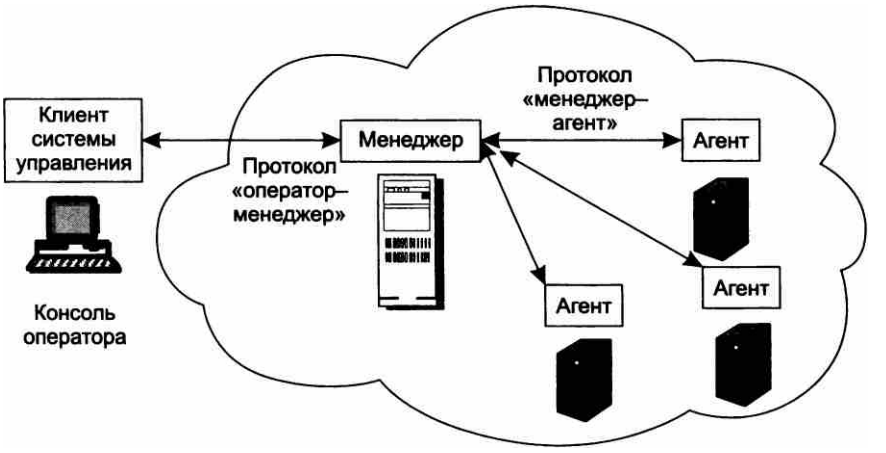


Рис. 25.2. Трехзвенная схема управления сетью

- Третье звено, *агент*, устанавливается на управляемом объекте или связанном с ним компьютере.

Взаимодействие менеджера, агента и управляемого объекта

Остановимся подробнее на той части системы управления, которая относится к взаимодействию менеджера, агента и управляемого объекта (рис. 25.3).

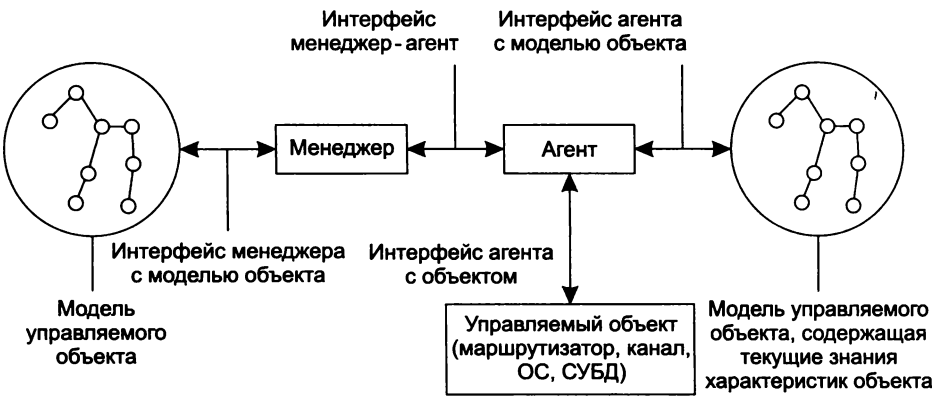


Рис. 25.3. Взаимодействие агента, менеджера и управляемого объекта

Для каждого управляемого объекта в сети создается некоторая *модель объекта*. Она представляет все характеристики объекта, которые нужны для его контроля. Например, модель маршрутизатора обычно включает такие характеристики, как количество портов, их тип, таблица маршрутизации, количество кадров и пакетов протоколов канального, сетевого и транспортного уровней, прошедших через эти порты. Модели объектов сети используются менеджером как источник знаний о том, какой набор характеристик имеет тот или иной объект.

Модель объекта совпадает с логической схемой базы данных (БД) объекта, хранящей значения его характеристик. Эта БД хранится на устройстве и постоянно пополняется результатами измерений характеристик, которые проводит агент.

В системах управления сетями, построенных на основе протокола SNMP, такая база называется **базой данных управляющей информации** (Management Information Base, **MIB**).

Менеджер не имеет непосредственного доступа к базе данных MIB, для получения конкретных значений характеристик объекта он должен по сети обратиться к его агенту. Таким образом, агент является посредником между управляемым объектом и менеджером. *Менеджер и агент взаимодействуют по стандартному протоколу*. Этот протокол позволяет менеджеру запрашивать значения параметров, хранящихся в MIB, а агенту — передавать информацию, на основе которой менеджер должен управлять объектом.

Различают *внутриполосное управление* (In-band), когда команды управления идут по тому же каналу, по которому передаются пользовательские данные, и *внеполосное управление* (Out-band), то есть осуществляемое вне канала передачи пользовательских данных.

Внутриполосное управление более экономично, так как не требует создания отдельной инфраструктуры передачи управляющих данных. Однако внеполосное управление надежнее, так как соответствующее оборудование может выполнять свои функции даже тогда, когда те или иные сетевые элементы выходят из строя и основные каналы передачи данных оказываются недоступными.

Схема «менеджер — агент — управляемый объект» позволяет строить достаточно сложные в структурном отношении системы управления.

Наличие нескольких менеджеров позволяет распределить между ними нагрузку по обработке управляющих данных, обеспечивая масштабируемость системы. Как правило, используются два типа связей между менеджерами: одноранговая (рис. 25.4) и иерархическая (рис. 25.5). Каждый агент, показанный на рисунках, управляет одним или несколькими элементами сети (Network Element, NE), параметры которых он помещает в соответствующую базу MIB. Менеджеры извлекают данные из баз MIB своих агентов, обрабатывают

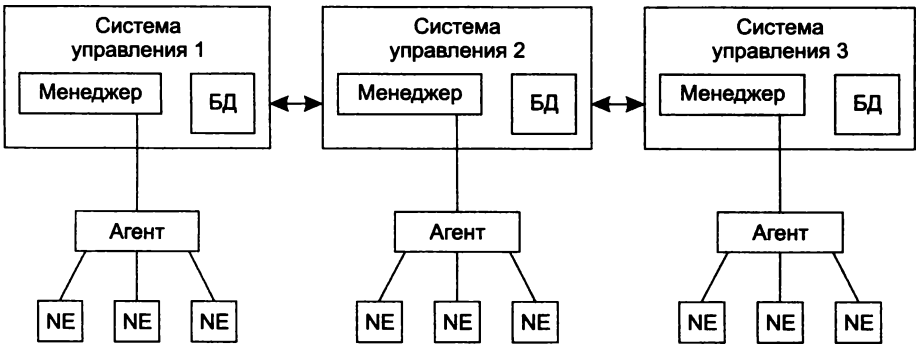


Рис. 25.4. Одноранговые связи между менеджерами

Рис. 25.5. Иерархические связи между менеджерами

их и хранят в собственных базах данных. Операторы, работающие за рабочими станциями, могут соединиться с любым из менеджеров и с помощью графического интерфейса просмотреть данные об управляемой сети, а также выдать менеджеру некоторые директивы по управлению сетью или ее элементами.

В случае *одноранговых* связей каждый менеджер управляет своей частью сети на основе информации, получаемой от нижележащих агентов. Центральный менеджер отсутствует. Координация работы менеджеров достигается за счет обмена информацией между базами данных менеджеров. Одноранговое построение системы управления сегодня считается неэффективным и устаревшим.

Значительно более гибким является *иерархическое* построение связей между менеджерами. Каждый менеджер нижнего уровня выполняет также функции агента для менеджера верхнего уровня. Подобный агент работает уже с укрупненной моделью MIB своей части сети. В такой базе MIB собирается именно та информация, которая нужна менеджеру верхнего уровня для управления сетью в целом.

Системы управления сетью на основе протокола SNMP

Протокол SNMP

Протокол SNMP (Simple Management Network Protocol — простой протокол сетевого администрирования) используется в качестве стандартного протокола взаимодействия менеджера и агента.

Протокол SNMP относится к прикладному уровню стека TCP/IP. Для транспортировки своих сообщений он использует дейтаграммный транспортный протокол UDP, который, как известно, не обеспечивает надежной доставки. Протокол TCP, организующий надежную передачу сообщений на основе соединений, весьма значительно загружает управляемые устройства, которые на момент разработки протокола SNMP были не очень мощными, поэтому от услуг протокола TCP было решено отказаться.

SNMP — это протокол типа «запрос-ответ», то есть на каждый запрос, поступивший от менеджера, агент должен передать ответ. Особенностью протокола является его чрезвычайная простота — он включает в себя всего несколько команд.

- Команда `GetRequest` используется менеджером для запроса агента о значении какой-либо переменной по ее стандартному имени.
- Команда `GetNextRequest` применяется менеджером для извлечения значения следующего объекта (без указания его имени) при последовательном просмотре таблицы объектов.
- С помощью команды `Response` SNMP-агент передает менеджеру ответ на команду `GetRequest` или `GetNextRrequest`.
- Команда `SetRequest` позволяет менеджеру изменять значения какой-либо переменной или списка переменных. С помощью команды `SetRequest` и происходит собственно управление устройством. Агент должен «понимать» смысл значений переменной, которая используется для управления устройством, и на основании этих значений

выполнять реальное управляющее воздействие: отключить порт, приписать порт определенной линии VLAN и т. п. Команда `Set` пригодна также для задания условия, при выполнении которого SNMP-агент должен послать менеджеру соответствующее сообщение. Таким образом может быть определена реакция на такие события, как инициализация агента, рестарт агента, обрыв связи, восстановление связи, неверная аутентификация и потеря ближайшего маршрутизатора. Если происходит любое из этих событий, то агент инициализирует прерывание.

- Команда `Trap` используется агентом для сообщения менеджеру о возникновении особой ситуации.
- Команда `GetBulk` позволяет менеджеру получить несколько переменных за один запрос.

SNMP-сообщения, в отличие от сообщений многих других коммуникационных протоколов, не имеют заголовков с фиксированными полями. Любое SNMP-сообщение состоит из трех основных частей: *версии протокола*, *общей строки* и *области данных*.

Общая строка (community string) используется для группирования устройств, управляемых определенным менеджером. Общая строка является своего рода паролем, так как для того, чтобы устройства могли взаимодействовать по протоколу SNMP, они должны иметь одно и то же значение этого идентификатора (по умолчанию часто употребляется строка «public»). Однако этот механизм служит скорее для «распознавания» партнеров, нежели для безопасности[1].

В области данных содержатся описанные команды протокола, а также имена объектов и их значения. Область данных состоит из одного или более блоков, каждый из которых может относиться к одному из перечисленных типов команд протокола SNMP. Для каждого типа команды определен свой формат. Например, формат блока, относящегося к команде `GetRequest`, включает следующие поля:

- идентификатор запроса;
- статус ошибки (есть или нет);
- индекс ошибки (тип ошибки, если она есть);
- список имен объектов SNMP MIB, включенных в запрос.

База данных MIB

База данных MIB содержит значения множества различных типов переменных, характеризующих конкретный управляемый объект. В самой первой версии стандарта (MIB-I) для характеристики устройства предлагалось использовать 114 типов переменных. Эти переменные организованы в виде дерева. Из корня выходит 8 ветвей, соответствующих следующим восьми группам переменных:

- *System* — общие данные об устройстве (например, идентификатор поставщика, время последней инициализации системы);
- *Interfaces* — параметры сетевых интерфейсов устройства (например, их количество, типы, скорости обмена, максимальный размер пакета);

---

[1] В версии SNMPv3 предусмотрены средства обеспечения конфиденциальности, целостности и аутентичности при обмене менеджера и агента данными.

- *Address Translation Table* — описание соответствия между сетевыми и физическими адресами (например, по протоколу ARP);
- *Internet Protocol* — данные, относящиеся к протоколу IP (адреса IP-шлюзов, хостов, статистика о IP-пакетах);
- *ICMP* — данные, относящиеся к протоколу ICMP;
- *TCP* — данные, относящиеся к протоколу TCP (число переданных, принятых и ошибочных TCP-сообщений);
- *UDP* — данные, относящиеся к протоколу UDP (число переданных, принятых и ошибочных UPD-дейтаграмм);
- *EGP* — данные, относящиеся к протоколу EGP (число принятых с ошибками и без ошибок сообщений).

Каждая группа характеристик образует отдельное поддерево. Далее приведены переменные поддерева переменных *Interfaces*, используемые для описания интерфейса управляемого устройства:

- `ifType` — тип протокола, который поддерживает интерфейс (эта переменная принимает значения всех стандартных протоколов канального уровня);
- `ifMtu` — максимальный размер пакета сетевого уровня, который можно послать через этот интерфейс;
- `ifSpeed` — пропускная способность интерфейса в битах в секунду;
- `ifPhysAddress` — физический адрес порта (MAC-адрес);
- `ifAdminStatus` — желаемый статус порта (`up` — готов передавать пакеты, `down` — не готов передавать пакеты, `testing` — находится в тестовом режиме);
- `ifOperStatus` — фактический текущий статус порта, имеет те же значения, что и `ifAdminStatus`;
- `ifInOctets` — общее количество байтов, принятое данным портом, включая служебные, с момента последней инициализации SNMP-агента;
- `ifInUcastPkts` — количество пакетов с индивидуальным адресом интерфейса, доставленных протоколу верхнего уровня;
- `ifInNUcastPkts` — количество пакетов с широковещательным или групповым адресом интерфейса, доставленных протоколу верхнего уровня;
- `ifInDiscards` — количество корректных пакетов, принятых интерфейсом, но не доставленных протоколу верхнего уровня, скорее всего, из-за переполнения буфера пакетов или же по иной причине;
- `ifInErrors` — количество пришедших пакетов, которые не были переданы протоколу верхнего уровня из-за обнаружения в них ошибок.

Помимо переменных, описывающих статистику по входным пакетам, имеется аналогичный набор переменных, относящийся к выходным пакетам.

Еще более детальную статистику о работе сети можно получить с помощью расширения SNMP — протокола **RMON** (Remote Network MONitoring — дистанционный мониторинг сети). Системы управления, построенные на основе RMON, имеют такую же архитектуру, элементами которой являются менеджеры, агенты и управляемые объекты. Отличие состоит в том, что SNMP-системы собирают информацию только о событиях, происходящих

на тех объектах, на которых установлены агенты, а RMON-системы — также о сетевом трафике. С помощью RMON-агента, встроенного в коммуникационное устройство, можно провести достаточно детальный анализ работы сетевого сегмента. Собрав информацию о наиболее часто встречающихся типах ошибок в кадрах, а затем получив зависимость интенсивности этих ошибок от времени, можно сделать некоторые предварительные выводы об источнике ошибочных кадров и на этом основании сформулировать более тонкие условия захвата кадров со специфическими признаками, соответствующими выдвинутой версии. Все это помогает автоматизировать поиск неисправностей в сети.

Режим удаленного управления и протокол telnet

Режим удаленного управления, называемый также режимом **терминального доступа**, предполагает, что пользователь превращает свой компьютер в виртуальный терминал другого компьютера, к которому он получает удаленный доступ.

В период становления компьютерных сетей, то есть в 70-е годы, поддержка такого режима была одой из главных функций сети. Устройства PAD сетей X.25 существовали именно для того, чтобы обеспечить удаленный доступ к мейнфреймам для пользователей, находившихся в других городах и работавших за простыми алфавитно-цифровыми терминалами.

Режим удаленного управления реализуется специальным протоколом прикладного уровня, работающим поверх транспортных протоколов, которые связывают удаленный узел с компьютерной сетью. Существует большое количество протоколов удаленного управления, как стандартных, так и фирменных. Для IP-сетей наиболее старым протоколом этого типа является telnet (RFC 854).

Протокол **telnet** работает в архитектуре клиент-сервер, он обеспечивает эмуляцию алфавитно-цифрового терминала, ограничивая пользователя режимом командной строки.

При нажатии клавиши соответствующий код перехватывается клиентом telnet, помещается в TCP-сообщение и отправляется через сеть узлу, которым пользователь хочет управлять. При поступлении на узел назначения код нажатой клавиши извлекается из TCP-сообщения сервером telnet и передается операционной системе узла. ОС рассматривает сеанс telnet как один из сеансов локального пользователя. Если ОС реагирует на нажатие клавиши выводом очередного символа на экран, то для сеанса удаленного пользователя этот символ также упаковывается в TCP-сообщение и по сети отправляется удаленному узлу. Клиент telnet извлекает символ и отображает его в окне своего терминала, эмулируя терминал удаленного узла.

Протокол telnet был реализован в среде Unix и наряду с электронной почтой и FTP-доступом к архивам файлов был популярным сервисом Интернета. Однако поскольку для аутентификации пользователей в технологии telnet применяются пароли, передаваемые через сеть в виде обычного текста, а значит, могут быть легко перехвачены и использованы, telnet сейчас работает преимущественно в пределах одной локальной сети, где возмож-

ностей для перехвата пароля гораздо меньше. Для удаленного управления узлами через Интернет вместо telnet обычно применяется протокол SSH (Secure SHell), который, так же как и telnet, был первоначально разработан для ОС Unix[1]. SSH, так же как и telnet, передает набираемые на терминале пользователя символы на удаленный узел без интерпретации их содержания. Однако в SSH предусмотрены меры по защите передаваемых аутентификационных и пользовательских данных.

Сегодня основной областью применения telnet является управление не компьютерами, а коммуникационными устройствами: маршрутизаторами, коммутаторами и хабами. Таким образом, он уже скорее не пользовательский протокол, а протокол администрирования, то есть альтернатива SNMP.

Тем не менее отличие между протоколами telnet и SNMP принципиальное. Telnet предусматривает обязательное участие человека в процессе администрирования, так как, по сути, он только транслирует команды, которые вводит администратор при конфигурировании или мониторинге маршрутизатора или другого коммуникационного устройства. Протокол SNMP, наоборот, рассчитан на автоматические процедуры мониторинга и управления, хотя и не исключает возможности участия администратора в этом процессе. Для устранения опасности, порождаемой передачей паролей в открытом виде через сеть, коммуникационные устройства усиливают степень своей защиты. Обычно применяется многоуровневая схема доступа, когда открытый пароль дает возможность только чтения базовых характеристик конфигурации устройства, а доступ к средствам изменения конфигурации требует другого пароля, который уже не передается в открытом виде.

Выводы

Система управления сетью — это сложный программно-аппаратный комплекс, который контролирует сетевой трафик и управляет коммуникационным оборудованием крупной компьютерной сети.

Наиболее распространенной является трехзвенная архитектура системы управления сетью, состоящая из администратора, программного менеджера и программного агента, встроенного в управляемое оборудование.

Для каждого управляемого объекта в сети создается некоторая модель объекта. Она представляет все характеристики объекта, которые нужны для его контроля.

Менеджер и агент работают на основе стандартных баз MIB, описывающих объекты управляемых коммуникационных устройств. Эта база данных хранится на устройстве и постоянно пополняется результатами измерений характеристик, которые проводит агент.

SNMP — это протокол стека TCP/IP, который организует взаимодействие между менеджером и агентом в режиме «запрос-ответ».

Режим удаленного управления, называемый также режимом терминального доступа, предполагает, что пользователь превращает свой компьютер в виртуальный терминал другого компьютера, к которому он получает удаленный доступ.

Режим удаленного управления реализуется специальным протоколом прикладного уровня, работающим поверх транспортных протоколов, которые связывают удаленный узел с ком-

[1] См. раздел «Аутентификация в ОС семейства Unix. Протокол SSH» в главе 27.

пьютерной сетью. Для IP-сетей наиболее старым протоколом этого типа является telnet, он обеспечивает эмуляцию алфавитно-цифрового терминала, ограничивая пользователя режимом командной строки.

Контрольные вопросы

1. Агент системы управления сетью — это:
   а) программа, устанавливаемая на управляемое устройство;
   б) аппаратный блок, встраиваемый производителем в управляемое устройство;
   в) аппаратный модуль, способный измерять характеристики проходящего через управляемое устройство трафика;
   г) программа, способная по командам выполнять конфигурирование управляемого устройства;
   д) программно-аппаратный модуль, управляющий вычислительным процессом в управляемом устройстве.
2. Что такое MIB в системе управления сетью? Варианты ответов:
   а) модель управляемого объекта;
   б) база данных управляющей информации;
   в) протокол взаимодействия агента и менеджера системы управления сетью;
   г) набор характеристик объекта, необходимых для его контроля.
3. Какую роль играет агент в двухзвенной схеме управления устройством? Варианты ответов:
   а) клиента;
   б) сервера;
   в) посредника.
4. Какой протокол может быть использован для взаимодействия менеджера и агента системы управления сетью? Варианты ответов:
   а) SNMP;
   б) SMTP;
   в) MIP;
   г) FTP;
   д) Telnet.

Часть VII

Безопасность компьютерных сетей

- Глава 26. Основные понятия, концепции и принципы информационной безопасности
- Глава 27. Технологии аутентификации, авторизации и управления доступом
- Глава 28. Технологии безопасности на основе фильтрации и мониторинга трафика
- Глава 29. Атаки на транспортную инфраструктуру сети
- Глава 30. Безопасность программного кода и сетевых служб

Эта часть книги открывается главой, в которой рассматриваются *фундаментальные* понятия, концепции и технологии, лежащие в основе любой системы информационной защиты: конфиденциальность, доступность, целостность, уязвимость, угроза, атака, ущерб, аутентификация, авторизация, контроль доступа. Эти понятия поясняются примерами некоторых распространенных пассивных и активных атак на транспортную систему и программное обеспечение сети: отказ в обслуживании, ответвление трафика, спуфинг, шпионы, кража личности. В качестве важнейшей концепции представлен системный подход к обеспечению безопасности — привлечение средств самой различной природы: законодательства, административных мер, процедур управления персоналом, средств физической защиты и программно-технического оборудования. Такой подход включает также следование универсальным принципам построения системы защиты: непрерывности и разумной достаточности, эшелонированного характера и сбалансированности. Завершает главу обзор криптографических методов, являющихся краеугольным камнем практически всех методов информационной безопасности.

Глава 27 посвящена *конкретным* технологиям аутентификации, авторизации и контроля доступа. Рассматриваются процедуры, основанные на одноразовых и многоразовых паролях, цифровых сертификатах и цифровой подписи, описываются мандатный, дескреционный и ролевой способы управления доступом. Приводятся примеры как локальных, так и централизованных систем аутентификации и авторизации, и в частности подробно обсуждается система Kerberos.

В главе 28 показано, как различные способы фильтрации могут быть использованы для экранирования сегментов сети от вредительского трафика. Рассматриваются системы мониторинга трафика на основе анализаторов протоколов и агентов протокола NetFlow, которые позволяют распознать атаку путем выявления отклонений образцов трафика от стандартного поведения. Изучаются различные типы файерволов: без запоминания состояния, с запоминанием состояния, с трансляцией адресов (NAT), с функцией прокси-сервера. Рассматриваются особенности анализа трафика и событий системами обнаружения вторжения (IDS). Дается обобщенная схема разбиения крупной корпоративной сети на зоны безопасности.

В главе 29 изучаются уязвимости и методы защиты транспортной инфраструктуры сети. Особое внимание уделяется безопасности службы DNS и протоколу маршрутизации BGP — двум очень важным элементам архитектуры Интернета, которые обеспечивают связность составляющих сетей и узлов в глобальном масштабе. Здесь также подробно рассматривается технология защищенного канала IPSec.

В завершающей главе изучаются типичные уязвимости программного обеспечения компьютерной сети, а также приводятся рекомендации по их устранению. Показаны различные методы обнаружения и обезвреживания вредоносного кода: троянских программ, червей, вирусов и программных закладок. Значительная часть этой главы посвящена вопросам безопасности сетевых служб — веб-службы, почты, облачных вычислений.

ГЛАВА 26 Основные понятия, концепции и принципы информационной безопасности

Идентификация, аутентификация и авторизация

Для пояснения таких базовых понятий информационной безопасности, как идентификация, аутентификация и авторизация, представим информационную систему (ИС) в виде упрощенной модели контролируемого доступа, когда несколько пользователей совместно работают с ресурсами информационной системы. Родившаяся полвека назад и направленная на повышение эффективности применения компьютера концепция разделения ресурсов выдвинула проблемы безопасности вычислительных систем на первый план — необходимо было *контролировать доступ* пользователей к компьютеру, защищая системные и пользовательские данные от ошибочных или злонамеренных действий. Контролируемый доступ является важным направлением обеспечения безопасности наряду с другими средствами безопасности: криптографической защитой, аудитом, сегментацией сети и т. п.

В модели контролируемого доступа определены объекты, субъекты, операции и система контроля доступа (рис. 26.1).

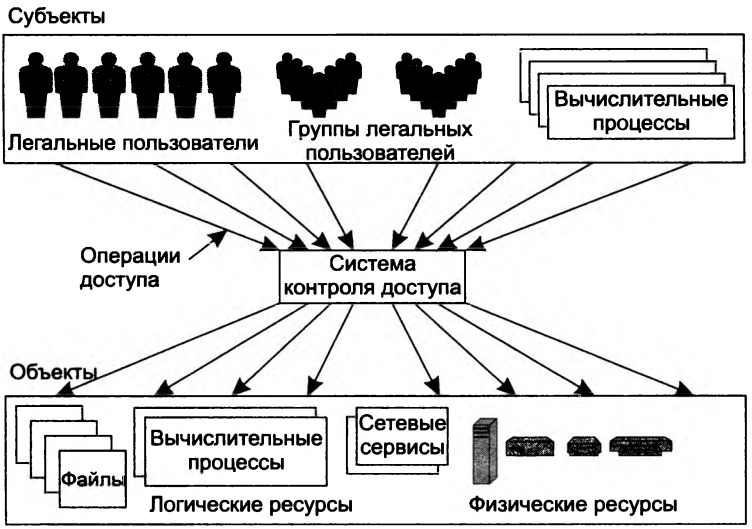


Рис. 26.1. Модель контролируемого доступа

Объекты представляют физические и логические информационные ресурсы ИС. К физическим ресурсам относятся как отдельные устройства целиком (процессор, внешние устройства, маршрутизаторы, коммутаторы, физические каналы связи и др.), так и физические разделяемые ресурсы устройств (разделы и секторы диска, процессорное время, физические соединения канала связи). Логическими ресурсами являются файлы, вычислительные процессы, сетевые сервисы, приложения, пропускная способность каналов связи и т. п.

Субъекты представляют сущности, между которыми разделяются информационные ресурсы. Это могут быть легальные пользователи ИС: персонал, поддерживающий работу ИС, внешние и внутренние клиенты; группы легальных пользователей, объединенные по различным признакам. Пользователь осуществляет доступ к объектам ИС не непосредственно, а с помощью прикладных процессов, которые запускаются от его имени. Поэтому в качестве субъектов выступают также прикладные вычислительные процессы. Иногда оказывается полезным представление в качестве субъектов системных вычислительных процессов.

Операции выполняются субъектами над объектами. Для каждого типа объектов существует собственный набор операций, которые с ними может выполнять субъект. Например, для файлов это операции чтения, записи, удаления, выполнения; для принтера — печать, перезапуск, очистка очереди документов, приостановка печати документа; для маршрутизатора — конфигурирование и т. д.

Система контроля доступа решает, какие операции разрешены для данного субъекта по отношению к данному объекту. Для автоматизированного контроля доступа необходимо, чтобы для каждой пары субъект-объект были однозначно определены *правила доступа*, на основании которых система могла бы разрешить или запретить выполнение каждой из предусмотренных для данного объекта операции.

Важнейшими элементами управляемого доступа являются процедуры идентификации, аутентификации и авторизации.

Идентификация — это присвоение объектам и субъектам информационной системы уникальных имен — **идентификаторов**.

Только при наличии уникальных идентификаторов система получает возможность распознавать и оперировать субъектами и объектами. Одни идентификаторы автоматически генерируются ОС и приложениями (идентификаторы процессов, идентификаторы логических сетевых соединений), другие назначаются администратором компьютерной сети (идентификаторы пользователей, адреса компьютеров, доменные имена сетевых сервисов), третьи порождаются обычными сетевыми пользователями, обладающими таким правом (выбор собственного имени, назначение имен файлам).

Идентификация пользователей представляет собой процедуру, выполняемую при логическом входе в систему, когда пользователь в ответ на выведенное на экране приглашение печатает свой идентификатор (имя), а система, сверяясь со своими данными, определяет, входит ли данное имя в число имен зарегистрированных (легальных) пользователей.

Пользователь может быть представлен в системе в виде нескольких субъектов и соответственно иметь несколько пользовательских идентификаторов. Например, один иден-

тификатор он может применять во время сетевой регистрации, а другой — для работы с корпоративной базой данных.

Аутентификация[1] — это процедура доказательства субъектом/объектом того, что он есть то, за что (кого) он себя выдает.

Аутентификация, или, другими словами, процедура установления подлинности, может применяться как к пользователям, так и к другим объектам и субъектам, в частности к данным, программам, приложениям, устройствам, документам (рис. 26.2).

Рис. 26.2. «В Интернете никто не узнает, что ты собака, если успешно пройдешь аутентификацию» (рисунок Питера Штайнера)

Аутентификация данных означает доказательство их подлинности, то есть того, что они поступили в неизмененном виде и именно от того человека, который объявил об этом.

В процедуре аутентификации участвуют две стороны:

- *аутентифицируемый* доказывает свою аутентичность, предъявляя некоторое доказательство — *аутентификатор*;
- *аутентифицирующий* проверяет эти доказательства и принимает решение.

Аутентификация бывает односторонней и двусторонней (взаимной).

Так, мы имеем дело с *односторонней аутентификацией*, в частности, при выполнении логического входа в защищенную систему. После того как пользователь сообщает системе свой идентификатор, он должен пройти процедуру аутентификации, то есть доказать, что именно ему принадлежит введенный им идентификатор (имя пользователя). Аутентификация предотвращает доступ к сети нежелательных лиц и разрешает вход для легальных пользователей.

---

[1] Термин «аутентификация» (authentication) происходит от латинского слова authenticus, которое означает подлинный, достоверный, соответствующий самому себе.

В качестве аутентификатора аутентифицируемый может продемонстрировать знание некоего общего для обеих сторон секрета, например слова (пароля), или обладание неким уникальным предметом (физическим ключом) либо предъявить собственные биохарактеристики (отпечатки пальцев).

В некоторых случаях односторонней аутентификации оказывается недостаточно и тогда используют *двустороннюю аутентификацию*. Например, пользователь, обращающийся с запросом к корпоративному веб-серверу, должен доказать ему свою легальность, но он также должен убедиться сам, что ведет диалог действительно с веб-сервером своего предприятия. Другими словами, сервер и клиент должны пройти процедуру взаимной аутентификации. Здесь мы имеем дело с двусторонней *аутентификацией на уровне приложений*. При установлении сеанса связи между двумя устройствами также часто предусматриваются процедуры взаимной *аутентификации устройств* на более низком, канальном, уровне.

Авторизация[1] — это процедура контроля доступа субъектов (пользователей, вычислительных процессов, устройств) к объектам (например, файлам, приложениям, сервисам, устройствам) и предоставления каждому из них именно тех прав, которые для них определены правилами доступа.

В отличие от аутентификации, которая позволяет распознать легальных и нелегальных пользователей, авторизация касается только *легальных* пользователей, успешно прошедших процедуру аутентификации.

Доступ к объектам, полученный в обход разрешений системы контроля доступа, называется *несанкционированным*, или *неавторизованным*.

Модели информационной безопасности

Понятие информационной безопасности может быть пояснено с помощью так называемых **моделей безопасности**. Суть этих моделей заключается в следующем: множество всех видов нарушений безопасности делится на несколько базовых групп таким образом, чтобы любое возможное нарушение обязательно можно было отнести по крайней мере к одной из этих групп. Затем система объявляется безопасной, если она способна противостоять каждой из этих групп нарушений.

Триада «конфиденциальность, доступность, целостность»

Одной из первых и наиболее популярных по сей день моделей безопасности является модель, предложенная Зальцером (Saltzer) и Шредером (Schroeder)[2]. Авторы постулировали, что все возможные нарушения информационной безопасности всегда могут быть отнесены

[1] Термин «авторизация» (authorization) происходит от латинского слова auctoritas, обозначавшего уровень престижа человека в Древнем Риме и соответствующие этому уровню привилегии.

[2] Jerry H. Saltzer, Mike D. Schroeder (September 1975), «The protection of information in computer systems».

по меньшей мере к одной из трех групп: нарушения конфиденциальности, нарушения целостности или нарушения доступности (рис. 26.3).

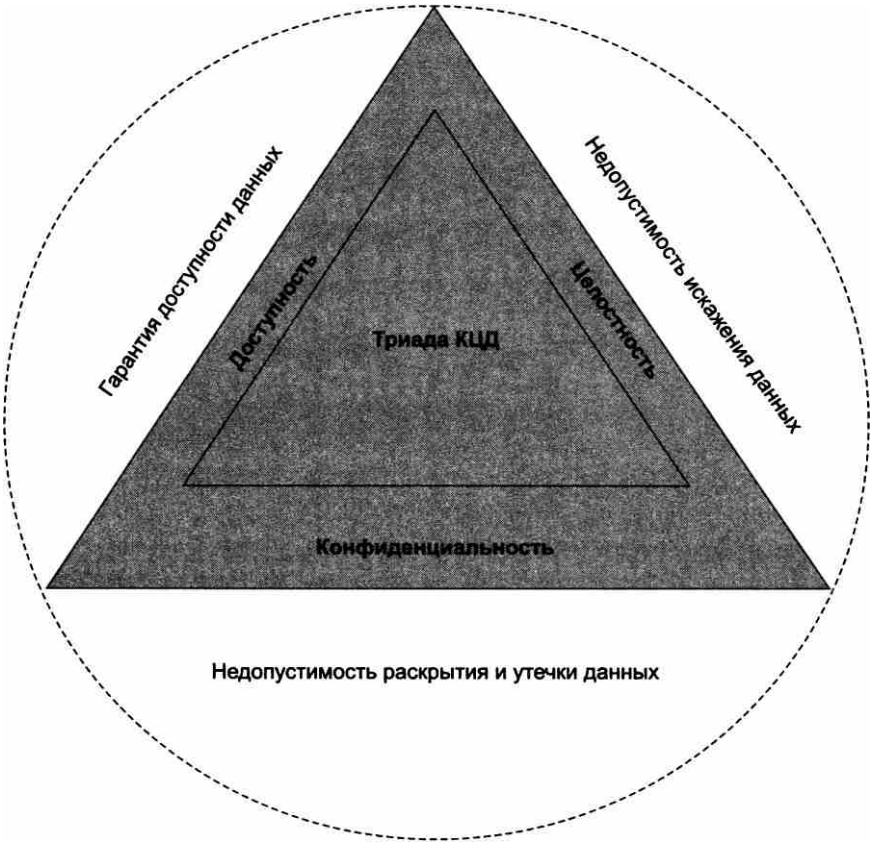


Рис. 26.3. Триада «конфиденциальность, целостность, доступность»

Соответственно информационная система находится в *состоянии безопасности*, если она защищена от нарушений конфиденциальности, целостности и доступности, где:

- **конфиденциальность** (confidentiality) — это состояние ИС, при котором информационные ресурсы доступны только тем пользователям, которым этот доступ разрешен;
- **целостность** (integrity) — это состояние системы, при котором информация, хранящаяся и обрабатываемая этой ИС, а также процедуры обработки информации не могут быть изменены, удалены или дополнены неавторизованным образом;
- **доступность** (availability) — это состояние системы, при котором услуги, оказываемые системой, могут гарантированно и с приемлемой задержкой быть предоставлены пользователям, имеющим на это право.

Для ссылки на триаду иногда используют аббревиатуру КЦД (*к*онфиденциальность, *ц*елостность, *д*оступность) или в англоязычной форме — CIA.

Требования к безопасности могут меняться в зависимости от назначения информационной системы, характера используемых данных и типа возможных угроз. Трудно представить систему, для которой нарушения целостности и доступности не представляли бы опасности, вместе с тем обеспечение конфиденциальности не всегда является обязательным.

Например, если вы публикуете информацию в Интернете на веб-сервере и вашей целью является сделать ее доступной для самого широкого круга людей, то конфиденциальность не требуется. Однако требования целостности и доступности остаются актуальными. Действительно, если вы не предпримете специальных мер по обеспечению целостности системы, то злоумышленник может изменить данные на вашем сервере и нанести этим ущерб вашему предприятию. Преступник может, например, внести изменения в помещенный на веб-сервере прайс-лист, что негативно отразится на конкурентоспособности вашего предприятия, или испортить коды свободно распространяемого вашей фирмой программного продукта, что, безусловно, скажется на ее деловой репутации. Если бы модифицированные данные были к тому же секретными, то в таком случае мы бы имели нарушение не только целостности, но и конфиденциальности.

Не менее важным в данном примере является и обеспечение доступности данных. Затратив немалые средства на создание и поддержание сервера в Интернете, предприятие вправе рассчитывать на отдачу: увеличение числа клиентов, количества продаж и т. д. Однако существует вероятность того, что злоумышленник предпримет атаку, в результате которой помещенные на сервер данные станут недоступными для тех, кому они предназначались. Примером таких злонамеренных действий может служить «бомбардировка» сервера пакетами, каждый из которых в соответствии с логикой работы соответствующего протокола вызывает тайм-аут сервера, что в конечном счете делает его недоступным для всех остальных запросов.

Некоторые виды нарушений безопасности могут быть приведены к модели КДЦ только путем расширительного толкования основополагающих понятий конфиденциальности, доступности и целостности. Так, свойство конфиденциальности по отношению, например, к устройству печати можно интерпретировать так, что доступ к устройству имеют те, и только те пользователи, которым этот доступ административно разрешен, причем они могут выполнять только те операции с устройством, которые для них определены. Свойство доступности устройства означает его готовность к работе всякий раз, когда в этом возникает необходимость. А свойство целостности может быть интерпретировано как свойство неизменности параметров данного устройства.

За 40 лет, прошедших с момента публикации статьи Зальцера и Шредера, информационные системы и среда, в которой они функционируют, претерпели революционные изменения, поэтому неудивительно, что появились новые типы нарушений, которые намного труднее (если вообще возможно) трактовать в терминах КДЦ. Рассмотрим, например, ситуацию, когда легальный клиент банка посылает по электронной почте запрос на снятие со счета крупной суммы, а затем заявляет, что этот запрос, который хотя и был послан от его имени, он не отправлял. Является ли это нарушением безопасности? Да. Были ли при этом нарушены конфиденциальность, доступность или целостность? Нет. Следовательно, список свойств безопасной системы следует расширить, добавив к КЦД еще одно свойство — «неотказуемость».

Неотказуемость (non-repudiation) — это такое состояние системы, при котором обеспечивается невозможность отрицания пользователем, выполнившим какие-либо действия, факта их выполнения, в частности отрицания отправителем информации факта ее отправления и/или отрицания получателем информации факта ее получения.

Гексада Паркера и модель STRIDE

Дискуссии о том, какой набор свойств ИС исчерпывающе характеризует ее безопасность, продолжаются, в результате предлагаются все новые и новые модели безопасности.

Одной из наиболее популярных альтернатив триаде КЦД является так называемая **гексада Паркера**[1] (Parkerian Hexad), в которой определено шесть базовых видов нарушений, в число которых, помимо нарушений конфиденциальности, доступности и целостности, входят еще три вида нарушений: аутентичности, владения и полезности (рис. 26.4).

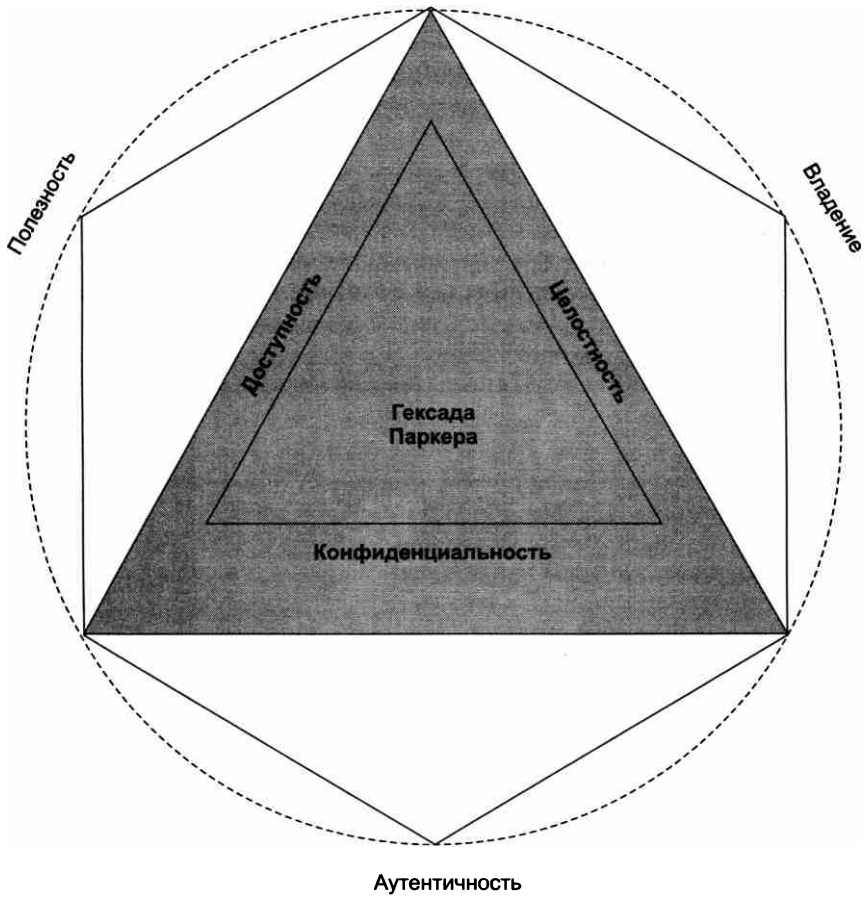


Рис. 26.4. Гексада Паркера

Аутентичность (authenticity) — это состояние системы, при котором пользователь не может выдать себя за другого, а документ всегда имеет достоверную информацию о его источнике (авторе). Из этого определения видно, что аутентичность является аналогом неотказуемости.

---

[1] Дон Паркер предложил свою гексаду в работе «Fighting Computer Crime» (1998).

Владение (possession) — это состояние системы, при котором физический контроль над устройством или другой средой хранения информации предоставляется только тем, кто имеет на это право.

Полезность (utility) — это такое состояние ИС, при котором обеспечивается удобство практического использования как собственно информации, так и связанных с ее обработкой и поддержкой процедур. В безопасной системе меры, предпринимаемые для защиты системы, не должны неприемлемо усложнять работу сотрудников, иначе они будут воспринимать их как помеху и пытаться при всякой возможности их обойти.

Еще одним вариантом определения безопасности ИС является модель **STRIDE**[1] (аббревиатура от англоязычных названий типов нарушений безопасности, перечисленных ниже). В соответствии с этой моделью ИС находится в безопасности, если она защищена от следующих типов нарушений: подмены данных, изменения, отказа от ответственности, разглашения сведений, отказа в обслуживании, захвата привилегий (рис. 26.5).

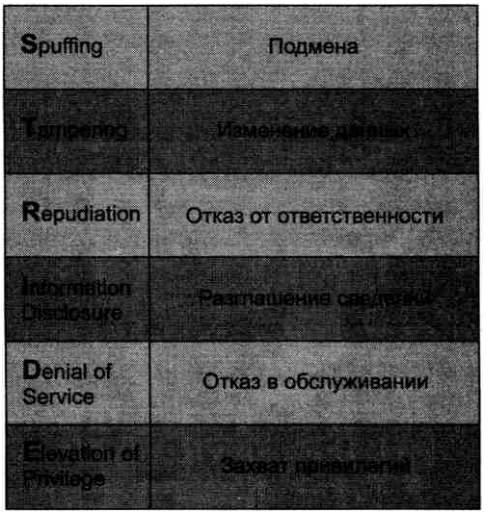


Рис. 26.5. Модель STRIDE

Подмена данных (spoofing) — это такое нарушение, при котором пользователь или другой субъект ИС путем подмены данных, например IP-адреса отправителя, успешно выдает себя за другого, получая таким образом возможность нанесения вреда системе.

Изменение (tampering) означает нарушение целостности.

Отказ от ответственности (repudiation) представляет собой негативную форму уже рассмотренного нами свойства неотказуемости (non-repudiation).

Разглашение сведений (information disclosure) — это нарушение конфиденциальности.

Отказ в обслуживании (denial of service) касается нарушения доступности.

---

[1] Модель STRIDE используется компанией Microsoft при разработке безопасного программного обеспечения.

Захват привилегий (elevation of privilege) заключается в том, что пользователь или другой субъект ИС несанкционированным образом повышает свои полномочия в системе, в частности незаконное присвоение злоумышленником прав сетевого администратора снимает практически все защитные барьеры на его пути.

Так же как и в гексаде Паркера, в модели STRIDE все возможное разнообразие нарушений безопасности сводится к шести типам нарушений, три из которых повторяют КЦД (с учетом того, что здесь эти три характеристики безопасности даны в негативном по отношению к КДЦ варианте), однако оставшиеся три — подмена данных, отказ от ответственности и захват привилегий — отличают модель STRIDE от гексады Паркера.

Российский государственный стандарт[1] дает определение информационной безопасности на основе гексады Паркера:

Информационная безопасность (*ИБ*) — [это] все аспекты, связанные с определением, достижением и поддержанием конфиденциальности, целостности, доступности, неотказуемости, подотчетности, аутентичности и достоверности информации или средств ее обработки.

Уязвимость, угроза, атака

Уязвимость (vulnerability) — это слабое звено информационной системы, которое, став известным злоумышленнику, может позволить ему нарушить ее безопасность.

Уязвимостями являются, например, ошибка в программе, примитивный пароль, неправильное назначение прав доступа к файлу с важными данными и множество других дефектов в разработке, эксплуатации или настройке системы.

Уязвимости системы могут быть скрытыми, то есть еще не обнаруженными, известными, но только теоретически, или же общеизвестными и активно используемыми злоумышленниками. Для общеизвестных уязвимостей в программных продуктах производители регулярно выпускают исправления, называемые **патчами** (patch — заплатка). Так, компания Microsoft даже назначила специальный день — каждый второй вторник каждого месяца, когда она объявляет о новых исправлениях в семействе ОС Windows. Многие из этих исправлений направлены на устранение уязвимостей. Однако к этой рутинной процедуре — регулярному внесению исправлений — не все и не всегда относятся с должным вниманием, из-за этого общеизвестные, но неисправленные ошибки в программном обеспечении являются одним из самых распространенных типов уязвимостей.

Другим типом уязвимостей, которыми часто пользуются злоумышленники, являются ошибки в конфигурировании программных и аппаратных средств. Например, имена «администратор» и «гость», установленные по умолчанию во многих ОС, могут облегчить злоумышленникам доступ к системе, поэтому они должны быть сразу при начальном конфигурировании ОС заменены другими, менее очевидными именами. С этой же целью администратор должен настроить подсистему интерактивного входа на то, чтобы она не показывала последнее имя пользователя, систему аудита — чтобы фиксировала все успешные и неуспешные попытки входа пользователей, а также выполнить другие столь же простые, но необходимые настройки.

---

[1] ГОСТ 13335-1:2006 «Информационная технология. Методы и средства обеспечения безопасности. Часть 1. Концепция и модели менеджмента безопасности информационных и телекоммуникационных технологий».

Поиск уязвимостей — важная часть задачи обеспечения безопасности. Эта работа включает в себя регулярное тестирование системы. В любой момент времени для любой системы можно указать множество различных видов уязвимостей, например, для операционных систем и приложений новые уязвимости появляются чуть ли не каждый день; выявлять их вручную — задача очень трудоемкая. Поэтому для автоматизации поиска уязвимостей используют различные программные инструменты — **средства сканирования уязвимостей**, такие, например, как McAfee, Nessus и др. Сканирование заключается в последовательном (адрес за адресом узла, или номер за номером порта, или идентификатор за идентификатором сетевого соединения) направлении запросов целевой системе. Затем на основании полученных ответов генерируется «информационный отпечаток» и, наконец, сравнением «отпечатка» с записями в базе данных выполняется идентификация уязвимости.

Другими базовыми понятиями информационной безопасности являются угроза и атака.

Угроза (threat) — набор обстоятельств и действий, которые *потенциально* могут привести к нарушению безопасности системы (то есть к нарушению ее конфиденциальности, целостности и доступности, если пользоваться моделью КЦД).

Атака (attack) — это реализованная угроза.

Мы в основном ограничимся рассмотрением только *технических* угроз, то есть угроз, исходящих из искусственно созданного человеком мира техники и технологий (в частности, из Интернета), не принимая во внимание угрозы, возникающие от природных катаклизмов, военных действий, террористических атак или экономических потрясений.

Атака может произойти только тогда, когда одновременно существуют уязвимость и направленная на использование этой уязвимости угроза (рис. 26.6).

То есть вполне возможна ситуация, когда система имеет некую уязвимость, но эта уязвимость еще не стала известной злоумышленникам — в данном случае соответствующая угроза отсутствует, а значит, и атака не может быть проведена. Аналогично, существование общеизвестной угрозы не влечет никакой опасности для системы, в которой нет соответствующей уязвимости. Например, появление информации о некоторой ошибке в коде ОС Windows может породить угрозу, но атака не осуществится, если эта уязвимость будет быстро устранена.

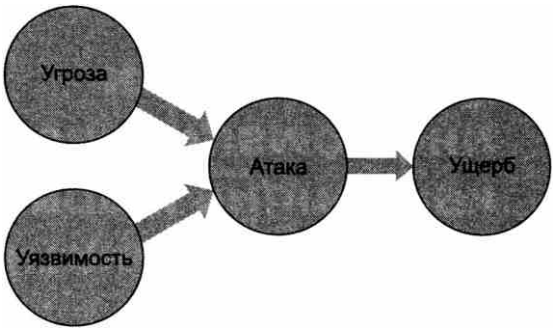


Рис. 26.6. Логическая связь между понятиями «уязвимость», «угроза», «атака», «ущерб»

Таким образом, любая угроза направлена на поиск и/или использование уязвимостей системы. В некоторых случаях злоумышленник работает «на ощупь», пытаясь обнаружить тот или иной дефект системы. Система реагирует на такого рода угрозы выдачей сообщений о мелких, но странных неполадках, а также флуктуациями в статистических характеристиках работы системы, на основании которых администратор сети или специалист по безопасности может заподозрить подготовку атаки.

Другие угрозы выражаются в четкой последовательности действий и имеют формализованное воплощение в виде **эксплойта** (exploit) — программы или просто последовательности командных строк, некоторой порции данных и/или пошаговом описании действий, которые, будучи выполненными, позволяют злоумышленнику воспользоваться некоторой конкретной уязвимостью информационной системы в своих интересах. Особая опасность эксплойта состоит в том, что, имея его в своем распоряжении, даже малоподготовленный хакер способен провести успешную атаку. Для этого ему достаточно зайти на один из многочисленных сайтов, снабжающих всех желающих своей «продукцией». Более того, в придачу к инструкциям и программам в Интернете можно найти даже предложения о сдаче в аренду целых **бот-сетей**[1], готовых к реализации мощных кибератак. В то же время, наличие у эксплойтов фиксированных признаков, таких, например, как специфические кодовые последовательности, облегчает распознавание и отражение соответствующих атак.

Угрозы могут исходить как от легальных пользователей сети, так и от внешних злоумышленников. В последние годы в статистике нарушений безопасности зафиксирован резкий сдвиг от внешних к внутренним угрозам. Примерно две трети от общего числа всех наиболее серьезных инцидентов, связанных с безопасностью, составляют нарушения или ошибки легальных пользователей сетей: сотрудников и клиентов предприятий, студентов, имеющих доступ к сети учебного заведения, и др.

Угрозы со стороны легальных пользователей могут быть как умышленными, так и неумышленными. К *умышленным* угрозам относятся, например, доступ и похищение конфиденциальных данных, мониторинг системы с целью получения информации о ее устройстве, посещение запрещенных веб-сайтов, вынос за пределы предприятия съемных носителей и т. п.

Безопасность может быть нарушена и в результате *непреднамеренных* нарушений пользователей и обслуживающего персонала — ошибок, приводящих к повреждению сетевых устройств, данных, программного обеспечения, ОС и приложений, беспечности в обеспечении секретности паролей и др. Известно, что правильное конфигурирование устройств является одним из мощных средств обеспечения безопасности. Но будучи выполненной с ошибками, эта операция способна обернуться своей противоположностью — угрозой. Как выяснилось, некоторые «атаки» на ИС были на самом деле не атаками, а ошибками администраторов сетей при выполнении конфигурирования элементов системы. Например, широко известен случай неверного конфигурирования протокола маршрутизации BGP в сети клиента провайдера AS7007, который привел к отказам работы большой части Интернета в 1997 году[2].

Угрозы внешних злоумышленников, называемых также **хакерами**, по определению являются умышленными и обычно квалифицируются как преступления. Среди внешних нарушителей безопасности встречаются люди, занимающиеся этой деятельностью профессионально или просто из хулиганских побуждений.

---

[1] Бот-сеть, или ботнет (botnet), — это организованная совокупность компьютеров, связанных через Интернет и способных согласованно решать задачи, поставленные перед ними злоумышленником.

[2] http://www.merit.edu/mail.archives/nanog/1997-04/msg00444.html.

Ущерб и риск. Управление рисками

Известно, что абсолютная безопасность информационной системы не может быть обеспечена никакими средствами: всегда есть вероятность появления ошибок и проведения новых атак со стороны злоумышленников. Поэтому целью обеспечения информационной безопасности является не исключение, а минимизация возможного негативного влияния, которое могут оказать на систему существующие угрозы. Из этого также следует, что надо каким-то образом ранжировать угрозы, чтобы решить, какими из них можно пренебречь, а на какие обратить основное внимание. Естественной мерой опасности атак и угроз является возможный ущерб, связанный с каждым из этих нарушений.

Ущерб (loss, impact) — это негативное влияние на систему, оказываемое проведенной атакой.

В качестве ущерба рассматриваются не только и не столько потери, связанные с восстановлением работы ИС, в частности серверов, файловой системы или системы аутентификации, — главное внимание должно быть уделено потерям, которые в результате этих нарушений понесло предприятие, строящее свой бизнес на базе этой ИС.

Важнейшей задачей обеспечения информационной безопасности является управление рисками. Здесь **риск** определяется как оценка ущерба от атаки с учетом вероятностной природы атаки. Другими словами, риск характеризуется парой:

{Ущерб от атаки, Вероятность атаки}.

Суть **управления рисками** — это системный анализ угроз, прогнозирование и оценка их последствий для предприятия, ранжирование угроз по степени их вероятного осуществления и опасности последствий и, наконец, выбор на приоритетной основе контрмер, направленных на смягчение или исключение возможного негативного воздействия этих нарушений на деятельность предприятия.

Управление рисками включает три укрупненных этапа (рис. 26.7):

1. Анализ уязвимостей.
2. Оценка рисков.
3. Управление рисками, или риск-менеджмент (принятие конкретных мер).

Анализ уязвимостей — объективное обследование реально существующих компьютерной сети, административных процедур и персонала. Угрозы определяются по отношению к *активам* предприятия, то есть ресурсам предприятия, представляющим для него ценность и являющимся объектом защиты (оборудование, недвижимость, транспортные средства, вычислительные устройства, ПО, документация и др.). Перечень угроз формулируется предположительно, то есть с использованием вероятностных категорий.

Оценка рисков — ранжирование возможных атак по степени опасности. Для этого вычисляются соответствующие риски — вероятностные оценки ущерба, который может быть нанесен предприятию каждой из атак в течение некоторого периода времени. Риск атаки тем выше, чем больше ущерб от нее и чем выше ее вероятность.

Риск-менеджмент — по каждому риску предпринимаются меры из следующего списка:

- *Принятие риска*. Этот вариант касается неизбежных атак, наносящих приемлемый ущерб.

- *Устранение риска.* Данный вариант имеет место, когда существующий риск можно свести на нет устранением либо уязвимости (например, сделать код коммерческого программного продукта открытым), либо угрозы (установить антивирусную систему).
- *Снижение риска.* Если риск невозможно ни принять, ни устранить, предпринимаются действия по его снижению. Например, всегда существует некоторая вероятность проникновения злоумышленников в систему путем подбора паролей. В таком случае риск несанкционированного доступа можно снизить, установив более строгие требования к длине и сменяемости паролей.
- *Перенаправление риска.* Если риск невозможно ни принять, ни устранить, ни даже существенно снизить, то риск может быть перенаправлен страховой компании.

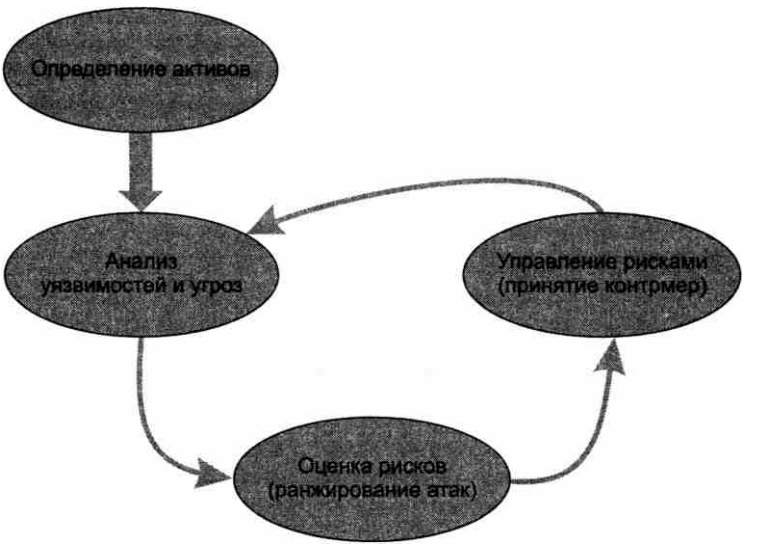


Рис. 26.7. Управление рисками

Типы и примеры атак

Пассивные и активные атаки

Атаки разделяют на активные и пассивные.

Активные атаки включают явные воздействия на систему, изменяющие ее состояние. Это могут быть зловредный программный код-вирус, внедренный в исполняемую системой программу, искажения данных на страницах взломанного веб-сайта, блокировка сетевого сервиса путем бомбардировки его ложными запросами или внедренное в коммуникационный протокол ложное сообщение. Главной отличительной чертой активных атак является то, что после своего завершения они, как правило, оставляют следы.

Многие активные кибератаки относят к типу *взламывания* (breaking-in) по аналогии с бытовыми ограблениями со взломом, когда хозяин заходит в свой дом и сразу обнаруживает

поврежденные замки, опустошенные ящики и разбросанные на полу вещи. В компьютерной системе после активного проникновения злоумышленника тоже остаются следы «взлома»: например, изменяется содержимое памяти, поступают странные диагностические сообщения, приложения начинают выполняться неправильно, замедленно или вообще зависают, в характеристиках сетевого трафика и в других статистических данных о работе системы появляются необъяснимые всплески активности.

Заметим, что иногда грабитель так хорошо «заметает следы», что пострадавший может сразу и не заметить преступления, особенно если он не обладает наблюдательностью Шерлока Холмса или Эркюля Пуаро. Так и в информационной системе тщательно подготовленная активная атака может пройти незамеченной, если специалисты, отвечающие за ее безопасность, плохо осведомлены о возможных последствиях такого рода атак.

Пассивные атаки не нарушают нормальную работу ИС, они связаны со сбором информации о системе, например прослушиванием внутрисетевого трафика или перехватом сообщений, передаваемых по линиям связи. Во многих случаях пассивные атаки не оставляют следов, поэтому их очень сложно выявить, часто они так и проходят незамеченными. Если использовать военную аналогию, то это разведка (но не боем).

Противопоставление активной и пассивной форм атак является некоторой идеализацией. На практике мы редко имеем дело с активной или пассивной атакой «в чистом виде». Чаще всего атака включает подготовительный этап сбора информации об атакуемой системе, а затем на основе собранных данных осуществляется активное вмешательство в ее работу. К полезной для хакера информации относятся типы ОС и приложений, IP-адреса, номера портов, имена и пароли пользователей. Часть информации такого рода может быть получена при анализе открытой информации или простом общении с персоналом (это называют **социальным инжинирингом**), а часть — с помощью тех или иных программ. В последнем случае мы сталкиваемся с другой последовательностью этапов: сначала выполняется активная фаза внедрения на атакуемый компьютер подслушивающей программы, затем период пассивного сбора информации (например, паролей пользователей), а затем снова активная фаза проникновения в компьютер.

Сейчас мы коротко, ограничиваясь обсуждением общей идеи, рассмотрим несколько типов популярных атак: отказ в обслуживании, спуфинг, внедрение кода, кража личности, фишинг, сетевая разведка. Более подробно эти, а также иные типы атак описаны в следующих главах.

Отказ в обслуживании

К числу активных атак относятся две весьма распространенные атаки: отказ в обслуживании и распределенная атака отказа в обслуживании.

Смысл атаки **отказа в обслуживании** (Denial of Service, DoS) прямо следует из ее названия. Система, предназначенная для выполнения запросов легальных пользователей, вдруг перестает это делать или делает с большими задержками, что эквивалентно отказу. Очевидный пример такой системы — веб-сайт. Наверняка 17 млн британских болельщиков Энди Марри «обрушили» бы сайт BBC, если бы трансляция финального теннисного матча Уимблдона в 2013 году шла только в Интернете (к счастью, параллельно шла телевизионная передача). Такие всплески запросов являются экстраординарными, и правильно спроектированные серверы справляются с нагрузкой, на которую они рассчитаны. Однако

отказ в обслуживании может наступить в результате не только резкой флюктуации интенсивности запросов, но и злонамеренных действий, когда перегрузка создается искусственно, а именно: на атакуемый компьютер посылается интенсивный поток запросов, сгенерированных средствами атакующего компьютера (рис. 26.8). Этот поток «затопляет» атакуемый компьютер, вызывая его перегрузку, и в конечном счете делает его недоступным. Блокировка происходит в результате исчерпания ресурсов либо процессора, либо операционной системы, либо канала связи (полосы пропускания).

При DoS-атаке используется тот простой факт, что компьютер подключен к сети — именно это является в данном случае уязвимостью.

К сожалению, для большинства современных пользователей устранить эту уязвимость простым отключением компьютера от Интернета нельзя, хотя в некоторых случаях, требующих особо высокого уровня безопасности, так и поступают.

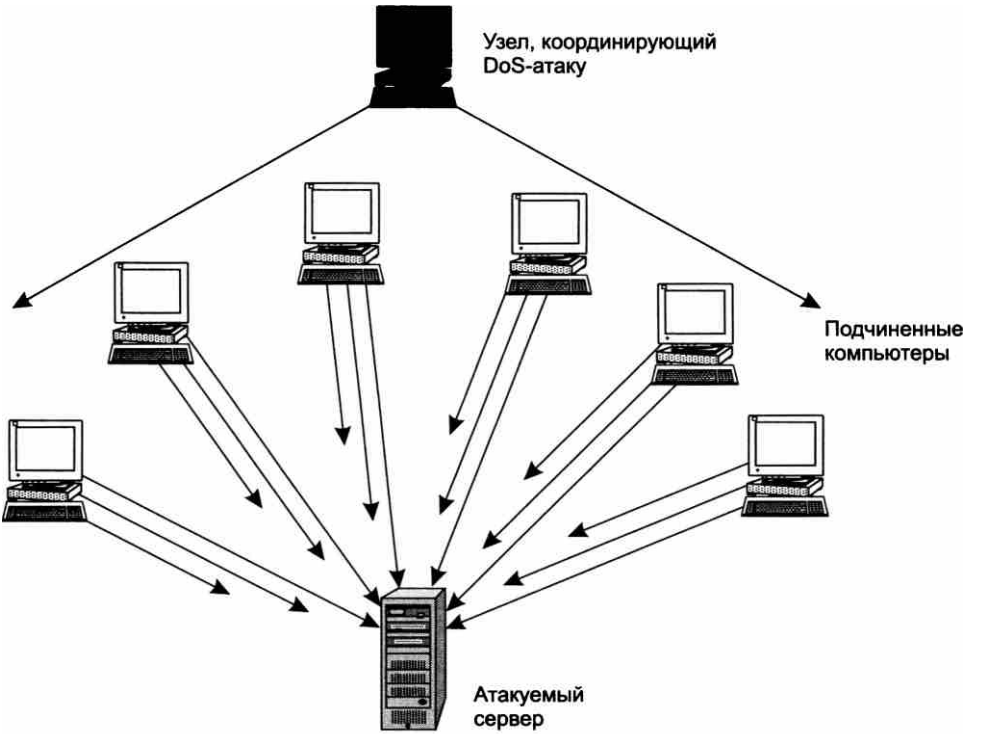


Рис. 26.8. Схема DDoS-атаки

Злоумышленник может многократно усилить эффект от проведения атаки отказа в обслуживании путем кражи чужой вычислительной мощности. Для этого он получает контроль над атакуемым компьютером, загружает в него вредительское программное обеспечение и активирует его. Таким образом злоумышленник незаметно для владельца «ответвляет» часть вычислительной мощности, заставляя компьютер работать на себя. При этом

владельцу компьютера не наносится никакого другого вреда, кроме снижения производительности его компьютера. Для проведения мощной атаки злоумышленник захватывает контроль над некоторым множеством компьютеров, организует их согласованную работу и направляет суммарный многократно усилившийся поток запросов с множества компьютеров-«зомби» на компьютер-жертву. Говорят, что в таких случаях имеет место **распределенная атака отказа в обслуживании** (Distributed Denial of Service, DDoS), или *DDoS*-атака.

При проведении атак злоумышленнику важно не только добиться своей цели, заключающейся в причинении ущерба атакуемому объекту, но и уничтожить все следы своего участия в этом. Одним из основных приемов, используемых злоумышленниками для «заметания следов», является *подмена содержимого пакетов*, или **спуфинг** (*spoofing*). В частности, для сокрытия места нахождения источника вредительских пакетов (например, при атаке отказа в обслуживании) злоумышленник изменяет значение поля адреса отправителя в заголовках пакетов. Поскольку адрес отправителя генерируется автоматически системным программным обеспечением, злоумышленник вносит изменения в соответствующие программные модули так, чтобы они давали ему возможность отправлять со своего компьютера пакеты с любыми IP-адресами. Еще труднее определить адрес источника распределенной атаки, так как непосредственными исполнителями выступают «зомбированные» компьютеры и именно их адреса содержатся в поле адреса отправителя пакетов, бомбардирующих компьютер-жертву. И хотя ничего не подозревающие владельцы компьютеров-исполнителей становятся участниками распределенной атаки помимо своей воли, большая часть ответственности ложится и на них. Ведь именно их недоработки в деле обеспечения безопасности собственных систем сделали возможной эту атаку.

Внедрение вредоносных программ

Многочисленная группа активных атак связана с внедрением в компьютеры **вредоносных программ** (malware — сокращение от malicious software). К этому типу программ относятся троянские и шпионские программы, руткиты, черви, вирусы, спам, логические бомбы и др. (рис. 26.9).

Рис. 26.9. Вредоносные программы

Эти программы могут проникать на атакуемые компьютеры разными путями. Самый простой из них — «самодоставка», когда пользователь загружает файлы из непроверенных источников (съемных носителей или веб-сайтов) либо беспечно открывает подозрительный файл, пришедший к нему как приложение по электронной почте. Существуют и более сложные представители вредоносных программ, обладающие собственными механизмами «размножения», копии таких программ распространяются по компьютерам сети без участия пользователей.

Одним из примеров вредоносных программ являются **шпионские программы** (spyware), которые тайно (как правило, удаленно) устанавливаются злоумышленниками на компьютеры ничего не подозревающих пользователей, чтобы отслеживать и фиксировать все их действия. В число таких действий может входить введение имени и пароля во время логического входа в систему, посещение тех или иных веб-сайтов, обмен информацией с внешними и внутренними пользователями сети и пр. Собранная информация пересылается злоумышленнику, который применяет ее в преступных целях.

Заметим, что в качестве шпионских программ могут использоваться не только созданные специально для этих целей вредоносные программы, но и программы легального назначения. Так, опасным средством шпионажа могут стать легальные системы мониторинга сети, такие, например, как популярные сетевые мониторы Wireshark или Microsoft Network Monitor. Исходное назначение этих программ состоит в том, чтобы дать администратору сети возможность следить за сетевым трафиком, в частности захватывать пакеты, используя механизм фильтрации, просматривать их содержимое, собирать статистику по загрузке устройств. В руках злоумышленника такая программа превращается в мощный инструмент взлома сети, который позволяет перехватывать пакеты с паролями и другой секретной информацией.

Потери, вызванные вредоносными программами, могут заключаться не только в уничтожении, искажении или похищении информации, приведении в нерабочее состояние программного обеспечения, а значит, и компьютера в целом, но и в значительных затратах времени и сил администраторов на обнаружение и распознавание атак, фильтрацию внешних сообщений, тестирование и перезагрузку систем. Вредоносные программы в начале этого десятилетия были одной из основных причин нарушения безопасности компьютерных сетей. Однако, как показала статистика, в последние два года суммарный ущерб, нанесенный вредоносными программами предприятиям, резко снизился. Это связывают в том числе с улучшением качества антивирусных средств и ужесточением наказания за такого рода преступления.

Кража личности, фишинг

По мере развития услуг, оказываемых через Интернет, все более популярными становятся аферы, когда один человек выдает себя за другого. Действительно, ведь в этом случае не требуется личное присутствие в офисе, и индивидуум доказывает свою идентичность, передавая обслуживающему центру свои персональные данные по телефону или используя интерактивную систему веб-сайта. Злоумышленник решает выдать себя за другого, чтобы, например, взять кредит на чужое имя, получить доступ к чужому счету, рассчитаться за покупку чужой карточкой, получить именное приглашение на закрытое мероприятие. Такую пассивную атаку, заключающуюся в сборе данных о другом человеке, называют **кражей личности** (identity theft).

Фишинг (phishing — искаженное fishing) используется мошенниками для «выуживания» персональных данных. Вы можете ответить на телефонный звонок, и человек, представившийся сотрудником банка или государственной налоговой службы, работником ЖКХ или представителем провайдера мобильной связи, начинает выспрашивать критичные данные о вас. Угроза может прийти и по электронной почте. Будущая жертва получает сообщение, в котором, к примеру, сообщается о якобы произведенной ею покупке, как правило, достаточно дорогой. Далее говорится, что при снятии средств за эту покупку у банковской системы возникли некие проблемы. Для разрешения ситуации клиенту предлагается срочно пройти по ссылке на сайт банка. Жертва, взволнованная тем, что никакой такой покупки она не совершала, торопится прояснить ситуацию, щелкает на предложенной ссылке и видит на экране знакомый логотип своего банка и интерактивную форму, запрашивающую персональные данные клиента, ИНН, номер счета, девичью фамилию матери и другие данные, которые нужны злоумышленнику.

Чем больше людей узнает о приемах выуживания информации, тем более изощренные методы обмана применяют преступники. Они создают поддельные сайты, выглядящие совсем как настоящие, они используют доменные имена, очень похожие на настоящие. Когда вам предлагают посетить сайт международной платежной системы PayPal, а в адресной строке браузера появляется адрес www.peypal.com, вы можете и не заметить подмены. Когда же наученные горьким опытом пользователи Интернета стали более внимательными, мошенники научились, используя несовершенства браузеров, помещать в поле адресной строки браузера имя настоящего сайта, в нашем случае — paypal.com. В такой ситуации даже самый внимательный пользователь может потерять бдительность и перейти на подставной сайт. И хотя эта уязвимость браузера была вскоре устранена, расслабляться нельзя — преступники продолжают совершенствовать свои приемы фишинга.

Следующим изобретением стали всплывающие окна. Предположим, клиент получает доступ к сайту своего банка (действительному, не поддельному) в результате прохождения стандартной процедуры идентификации и аутентификации. Он просматривает страницы сайта — нет никаких сомнений, что это реальный сайт. В какой-то момент на экране появляется всплывающее окно, которое стилистически выглядит как неотъемлемая часть сайта. В этом окне размещена интерактивная форма, запрашивающая персональные данные. Клиент чувствует себя в полной безопасности и вводит все запрашиваемые критичные данные. Однако настоящий сайт банка является только фоном, на котором располагаются окна-ловушки злоумышленника.

Иерархия средств защиты от информационных угроз

Обычно первое, что ассоциируется с информационной безопасностью, — это антивирусные программы, файерволы, системы шифрования, аутентификации, аудита и другие технические средства защиты. Бесспорно, роль этих средств в обеспечении безопасности велика, однако не меньшее, а иногда и большее влияние на безопасность системы оказывают средства, построенные на качественно иной основе.

Видеокамера и надежный замок в офисе, продуманная процедура приема сотрудников на работу, закон, угрожающий хакеру уголовным преследованием, стандарт, помогающий провести анализ возможного ущерба из-за действия нарушителя, — все эти мало схожие между собой средства одинаково важны для обеспечения безопасности.

Успех в области информационной безопасности может принести только *системный подход*, при котором средства защиты разных типов применяются совместно и под централизованным управлением.

Общепризнанным является представление множества разных средств защиты в виде четырех иерархически организованных уровней, средства каждого из которых могут быть использованы на разных этапах жизненного цикла системы обеспечения информационной безопасности (рис. 26.10).

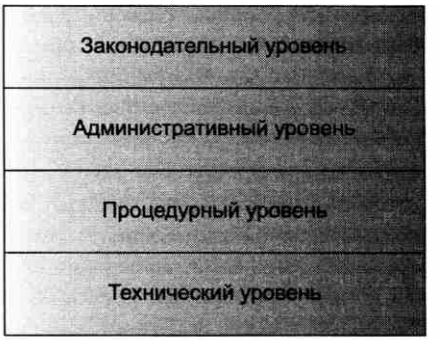


Рис. 26.10. Многоуровневая модель средств зашиты от информационных угроз

Средства безопасности законодательного уровня

К этому уровню средств безопасности относятся правовое регулирование, стандартизация, лицензирование и морально-этические нормы, принятые в обществе.

Законодательство может прямо влиять на концепцию построения защиты. Например, выход Закона РФ «О персональных данных», регламентирующего меры по обеспечению безопасности персональных данных при их обработке, потребовал от многих предприятий пересмотра и внесения принципиальных изменений в процедуры и инфраструктуру обработки информации. Не менее принципиальными для продолжения работ могут оказаться требования сертификации средств защиты данных, которые предполагается использовать в проектируемой системе обеспечения безопасности, или необходимость получения лицензии для выбранного вида деятельности.

Информационные технологии пронизывают нашу жизнь, поэтому совершенно естественно, что правоотношения в данной сфере уже давно регулируются развитым законодательством. Участниками правоотношений являются *субъекты правоотношений*, к которым в области информационной безопасности можно отнести обладателей и потребителей информации, владельцев интернет-сайтов, провайдеров телекоммуникационных сетей, разработчиков программных и аппаратных средств информационных технологий и др. (рис. 26.11).

Права и обязанности субъектов правоотношений определяются по отношению к *объектам правоотношений*, к которым принадлежат, например, ИС (государственные и частные), средства защиты информации (алгоритмы шифрования, аппаратные ключи, используемые для аутентификации, и т. п.), услуги, оказываемые в области ИБ, информация ограниченного доступа, в том числе информация, составляющая государственную, коммерческую, банковскую, семейную тайну, тайну переписки, персональные данные и др.

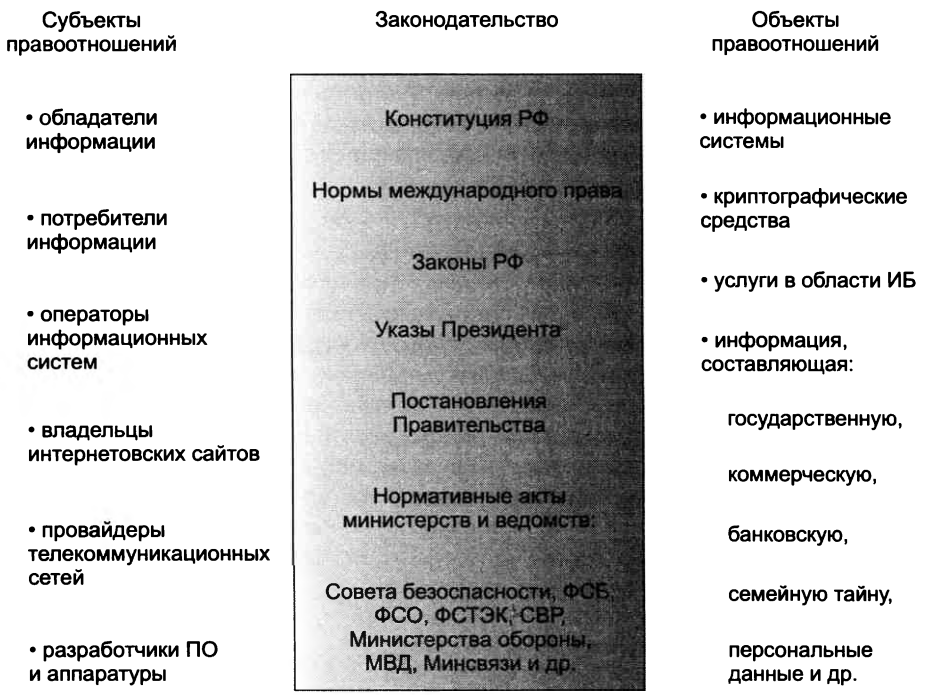


Рис. 26.11. Субъекты и объекты правоотношений в области информационной безопасности

Необходимость защиты информационных ресурсов и поддерживающей их инфраструктуры диктуется как нашими *правами* на защиту (например, каждому гражданину должна быть обеспечена защита от раскрытия его персональных данных), так и *обязанностями* ее защищать (каждая организация, оперирующая персональными данными, обязана защищать их от раскрытия).

Нормативно-правовой акт — это официальный документ, принятый компетентным правотворческим органом и устанавливающий общеобязательное государственное предписание, рассчитанное на многократное применение. Система нормативных правовых актов, действующих на территории страны, принятых законодательным (представительным) органом, называется **законодательством**.

Законодательство включает меры *ограничительно-репрессивного характера*, направленные на предотвращение нарушений, в том числе путем применения наказаний (например, уголовный кодекс), и меры *созидательного* характера, направленные на координацию работ в сфере ИБ, обучение и помощь в создании и использовании средств обеспечения информационной безопасности (например, стандарты).

К числу нарушений законодательства в области ИБ относят как традиционные «компьютерные» преступления, такие как нарушения доступности данных (DDoS-атаки), использование вредоносного ПО, превышение привилегий, несанкционированный доступ и др., так и нарушения регламентирующих правил, например отсутствие лицензии на определенный вид деятельности в области защиты информации, использование несертифицированных

продуктов там, где это требуется законом (к примеру, средств шифрования при работе с информацией, составляющей государственную тайну).

Определение нарушений и соответствующих наказаний в области ИБ можно найти в уголовном, семейном, гражданском кодексах, кодексе об административно-правовых нарушениях, а также в нормативных актах федеральных органов исполнительной власти.

К сожалению, правовые акты часто не успевают за стремительным развитием информационной сферы. В новых законах часто присутствует неопределенность того, когда следует применять специально предусмотренные наказания за киберпреступления, а когда — «традиционные» статьи уголовного кодекса. В некоторых случаях специалисты права, соглашаясь с общественным мнением, отмечали неадекватность компьютерных преступлений и соответствующих наказаний. Так, в 2012 году братья Попелыши посредством фишинга завладели банковскими данными граждан и совершили хищение на сумму в полмиллиона долларов. Однако в соответствии с нынешним законодательством их преступление не было квалифицировано как кража, так как электронные деньги по закону не являются объектом кражи. В результате первого в истории РФ дела о фишинге преступники были приговорены лишь к условным срокам за неправомерный доступ к компьютерной информации, а также за использование и распространение вредоносных программ.

Важным направлением законодательства в области безопасности является стандартизация.

Стандарты регулируют самые различные сферы и аспекты обеспечения информационной безопасности, в том числе теоретические концепции и алгоритмы, требования к программным и аппаратным средствам, методики обследования систем и порядок документирования результатов, административные процедуры. Стандартные процедуры оценки систем дают возможность их сопоставления и сравнения, на основании чего может выполняться **сертификация систем** на соответствие определенным требованиям. Стандарты безопасности определяют перечень тех свойств и функций, наличие которых является необходимым для того, чтобы та или иная система обрабатывала информацию безопасным образом.

К числу самых известных сертификационных стандартов относят **Оранжевую книгу**[1]. Этот самый заслуженный и популярный стандарт оценивает степень защищенности ОС, а также позволяет формализовать процедуру оценки, для чего в нем определяются формальные критерии отнесения системы к тому или иному классу безопасности. Впервые этот стандарт был опубликован в 1985 году в составе так называемой радужной серии стандартов информационной безопасности, издававшейся в период с 1980 по 1990 год под эгидой Министерства обороны США. Все 37 книг этой серии имели обложки разнообразных цветов. Именно цвет обложки и дал второе, неформальное, название «Оранжевая книга» стандарту «Критерии оценки доверенных компьютерных систем». Следует упомянуть также **Красную книгу** — еще один стандарт из «радужной» серии, который представляет собой интерпретацию Оранжевой книги для сетевых конфигураций. Однако оба этих стандарта не являются международными, к тому же их применимость ограничена только операционными системами. Эти ограничения в определенной степени снимает международный стандарт, известный под кратким (неофициальным) названием **Общие критерии** и позволяющий оценивать и сертифицировать программные продукты различных классов.

---

[1] Формальное название этого стандарта: «Министерство обороны США, Критерии оценки доверенных компьютерных систем» (Department of Defence, Trusted Computer System Evaluation Criteria).

Административный уровень. Политика безопасности

Основу административного уровня средств безопасности составляет **политика безопасности**, которая определяет стратегические направления информационной защиты предприятия, а именно очерчивает круг критически важных информационных ресурсов предприятия, защита которых представляет наивысший приоритет, предлагает возможные меры устранения или уменьшения связанных с этими ресурсами рисков. На основе найденной стратегии разрабатывается программа обеспечения безопасности ИС, планируется совокупный бюджет, необходимый для выполнения программы, назначаются руководители и ограничивается зона их ответственности.

В любом целенаправленном деле наличие достаточных материальных ресурсов — это только необходимое, но не достаточное условие; для достижения цели требуется выработать осмысленный план действий. Аналогично, при построении информационной защиты нельзя целиком полагаться на технические средства — никакие самые современные файерволы, системы обнаружения вторжений, сканеры уязвимостей, централизованные серверы аутентификации не защитят организацию, если не будет выработана руководящая идея, которая превращает набор отдельных мощных, но часто не слишком эффективных инструментов и методов в интегрированную систему, работающую на достижение общей цели. Если учесть, что для успешной реализации основополагающей идеи необходимо с самого начала располагать хотя бы самыми общими соображениями, в каком направлении двигаться, то мы приходим к понятию «политика».

Политика — это общее руководство, устанавливающее главные направления, в которых нужно двигаться, чтобы наиболее рациональным путем достичь поставленной цели. Содержание политики выражается в ее целях, программах и ценностях, в проблемах и задачах, которые она решает, в мотивах, механизмах, способах и методах принятия и реализации решений.

Сферой приложения политики может быть любая целенаправленная деятельность. Мы здесь рассматриваем *политику информационной защиты предприятия*, которую будем называть сокращенно *политикой безопасности* (ПБ). В сфере информационных технологий применяют политики и в других более узких сферах: например, политика информационной защиты компьютерных систем предприятия, политика использования средств коммуникаций, политика использования корпоративной электронной почты и т. п.

Как и любая политика, ПБ призвана играть организующую и дисциплинирующую роль. Процесс выработки ПБ приводит к более ясному осознанию целей и путей построения стратегии обеспечения информационной безопасности (СОИБ).

Политика безопасности разрабатывается с привлечением высшего руководства. Тем самым руководители демонстрируют свою поддержку предлагаемой стратегии обеспечения информационной безопасности, что имеет большое значение, так как ее реализация может потребовать привлечения значительных финансовых средств и ресурсов предприятия.

Будучи принятой, ПБ становится «законом», обязательным для исполнения всеми сотрудниками предприятия. Персонал предприятия должен быть ознакомлен с положениями ПБ, в том числе с ответственностью, которая определена за ее нарушения.

Политика безопасности фиксируется в документах. Часто под политикой понимают именно ее документальное выражение. Так, например, ГОСТ 50922-2006 дает следующее определение:

> **Политика безопасности**: совокупность документированных правил, процедур, практических приемов или руководящих принципов в области безопасности информации, которыми руководствуется организация в своей деятельности.

Решения, которые должны быть приняты в рамках разработки политики безопасности предприятия, а также документы, их описывающие, могут быть отнесены к верхнему, среднему или нижнему уровню.

Верхний уровень политики безопасности включает решения, затрагивающие предприятие в целом. Они принимаются высшим руководством, носят общий характер, могут быть описаны компактным документом. На этом уровне выполняется всестороннее обследование предприятия: выявляются критически важные активы, более всего нуждающиеся в защите, устанавливаются правила разграничения доступа к информационным ресурсам, определяются наиболее вероятные угрозы, оцениваются возможные потери, принимаются концептуальные решения относительно методов обеспечения защиты. То есть *значительная часть ПБ базируется на результатах анализа рисков*. На верхнем уровне принимаются организационно-административные решения, а именно: определяются должностные позиции (роли), создаются административные подразделения, комитеты, рабочие группы, функциями которых является проведение политики безопасности в жизнь, устанавливаются границы ответственности всех этих административных единиц. ПБ верхнего уровня может содержать указания на приверженность предприятия тем или иным стандартам и нормативно-правовым актам, принципы обучения персонала, порядок реагирования на нарушения режима безопасности и др.

К среднему уровню политики безопасности относят решения и соответствующие документы, касающиеся частных аспектов информационной безопасности, таких, например, как политика использования средств криптографической защиты, политика антивирусной защиты, политики мониторинга и менеджмента инцидентов информационной безопасности, политика защиты коммуникационных каналов, политика физической защиты и др. По сравнению с верхним уровнем разработка ПБ среднего уровня требует большего участия технических специалистов (из числа руководителей). Обычно в ПБ среднего уровня имеется упоминание о запрещенных действиях и наказаниях за них. Например, в рамках ПБ компьютерной инфраструктуры предприятия предусматриваются меры, защищающие сеть от вирусов и других вредоносных программ. Для этого в ПБ включаются пункты о запрете персоналу устанавливать и запускать ПО без его предварительного тестирования, о необходимости его регулярного обновления, о регламентации использования в корпоративной сети собственного оборудования сотрудников (ноутбуков, планшетов, портативных переносных USB-накопителей, флеш-карт) и другие подобные меры.

Политика безопасности *нижнего уровня* определяет действия по обеспечению безопасности на уровне сетевых сервисов и может представлять собой руководства, инструкции, регламенты и правила, связанные с администрированием и использованием сервисов. В отличие от двух верхних уровней, многие положения которых носят общий характер, на нижнем уровне приводятся более специфические, более детальные, более формализованные рекомендации, учитывающее особенности конкретного сервиса. Например, в приведенном примере ПБ среднего уровня в отношении защиты компьютерной сети от вирусов и других вредоносных программ среди прочего содержится требование о необходимости регулярно-

го тестирования программного обеспечения. Однако оно оставляет много вопросов. Какие методики должны использоваться при тестировании? Должны ли тестироваться программы, разработанные специалистами компании? Как надо интерпретировать результат тестирования? Ответы на подобные вопросы должны содержаться в ПБ нижнего уровня.

В то же время частные реализации политики безопасности нижнего уровня нельзя смешивать с обычными инструкциями. В первом случае объекты регламентации — процессы использования базы данных, файлового сервиса, корпоративной телефонной связи, электронной почты, антивирусных программ и т. д. и т. п. — оказывают непосредственное влияние на безопасность предприятия и поэтому должны разрабатываться централизованно, с участием руководства. Во втором случае инструкцию может составить любой специалист, обладающий достаточным для этого уровнем профессиональных знаний.

Каждый документ более высокого уровня раскрывается и дополняется одним или несколькими документами более низкого уровня. Чем выше уровень документа, тем более компактным и декларативным он является. Граница между уровнями является условной: так, например, документы среднего уровня ПБ, описывающие частные реализации политики, могут частично включать в себя политику защиты сервисов, то есть политику нижнего уровня.

Средства безопасности процедурного уровня

Средства безопасности процедурного уровня решают задачи, поставленные вышележащим административным уровнем, с использованием технических средств, предоставляемых нижележащим техническим уровнем. В качестве основного средства процедурного уровня выступает *человек*, выполняющий взаимосвязанную последовательность действий, направленную на решение той или иной задачи обеспечения безопасности.

Любой аспект информационной безопасности предполагает использование средств процедурного уровня. Даже простое поддержание нормального режима работы информационной системы осуществляется за счет выполнения множества повседневных процедур: резервного копирования, управления программным обеспечением, профилактических работ и т. п. Многие процедуры включают в себя применение технических средств. Например, задача учета различных ресурсов (документации, магнитных лент с резервированными данными, программ, оборудования и др.), как правило, решается с привлечением специально разработанных для этих целей программ. К средствам процедурного уровня относится также физическая защита: пропускной режим на территорию предприятия, охрана границ территории и др.

В общем случае *процедура* — это взаимосвязанная последовательность действий, направленная на решение некоторой задачи. Процедура может быть формальной, как, например, при ее представлении в виде алгоритма или программы на языке программирования, так и неформальной, в виде в той или иной меры расплывчатых указаний. Формальные процедуры могут быть реализованы механическими или электронными устройствами, но только человек может действовать в условиях, когда невозможно абсолютно однозначно определить все детали процедуры, а именно такова большая часть процедур безопасности, связанных с поддержанием работоспособности системы, управлением персоналом, физической защитой, управлением документацией и др.

Именно поэтому между уровнями стратегий безопасности и технических средств (способных реализовывать только формальные процедуры) с необходимостью появляется промежуточный уровень неформальных процедур, приводимых в действие человеком.

Исполнителями процедур являются ИТ-специалисты, сотрудники отделов информационной безопасности, пользователи и другие сотрудники, связанные с информационной защитой.

В качестве примера задачи, решаемой на процедурном уровне, рассмотрим управление персоналом.

Управление персоналом включает подбор персонала, прием на работу, увольнение, текущий контроль и др. Каждое из перечисленных действий имеет отношение к безопасности. При подборе работников следует проверять прошлое кандидатов, их рекомендации, в некоторых случаях требуется дополнительная проверка профессиональных сертификатов, кредитной истории, записей в базах данных о преступниках, другие более тщательные проверки.

В процедуре приема на работу должно быть предусмотрено ознакомление сотрудника с мерами ответственности за нарушения правил информационной безопасности. В частности, работник должен официально подтвердить свое согласие следовать политике безопасности предприятия и нести ответственность за ее нарушение, подписать соглашение о неразглашении конфиденциальных данных. Такой порядок помогает разрешать потенциальные конфликтные ситуации на правовой основе.

Особое внимание служба безопасности должна уделять процедуре увольнения, так как, по статистике, большое число нарушений информационной безопасности совершается как раз лицами, потерявшими работу. Каждый уволенный по негативным обстоятельствам представляет собой угрозу раскрытия конфиденциальной информации, к которой он имел допуск. Процедура увольнения должна включать немедленную блокировку всех учетных записей, смену паролей, блокировку удаленного доступа. Особенно серьезная угроза исходит от уволенного системного администратора, в таких случаях должна быть применена специальная процедура, включающая помимо обычных мер полный аудит системы.

Должны быть также предусмотрены процедуры текущего контроля сотрудников. Контроль включает процедуры отчетности в соответствии с административной субординацией, оценку работы (ежегодная аттестация сотрудников), продвижение по служебной лестнице. Рабочая, доброжелательная атмосфера на предприятии является важным фактором производительности и безопасности. Однако необходимо соблюдать разумный компромисс между культивируемой в большинстве компаний атмосферой непринужденности и доверия между сотрудниками, с одной стороны, и необходимой для обеспечения безопасности атмосферой взаимного контроля и подозрительности — с другой.

При управлении персоналом следует придерживаться нескольких общепризнанных принципов.

Разграничение обязанностей преследует несколько целей: во-первых, это способствует повышению производительности за счет специализации, во-вторых, устраняет ненужное дублирование, а в-третьих (что важно для безопасности), не дает концентрировать слишком много полномочий в одних руках. В некоторых особо критичных случаях такое разграничение вводится для того, чтобы некое действие могло быть произведено только с участием двух (или более) человек. Пример такой процедуры — доступ к банковской ячейке, которая открывается двумя ключами: ключом владельца содержимого ячейки и ключом представителя банка.

Правило обязательного отпуска, помимо заботы о здоровье сотрудника, дает возможность в его отсутствие основательно проверить, нет ли нарушений в его работе (одним из косвенных признаков этого может служить исчезновение какой-либо проблемы одновременно с его уходом в отпуск), а также не использует ли какой-либо злоумышленник его учетную

запись (доказательством этого служат записи в журнале регистрации событий, связанные с учетной записью сотрудника, после его ухода в отпуск).

Принцип минимально необходимого уровня привилегий означает, что каждый сотрудник должен иметь только тот тип доступа и только к тем ресурсам, которые ему необходимы для выполнения его служебных обязанностей. Нарушение этого принципа мы могли наблюдать в известном случае Брэдли Мэннинга, рядового американской армии, который, находясь на удаленной военной базе в Ираке, сумел передать WikiLeaks огромный объем секретных документов. Для того чтобы добыть эту информацию, Мэннинг использовал удаленный доступ к сертифицированной сети Министерства обороны США. И хотя Мэннинг и его сослуживцы имели допуск к секретным данным, те данные, которые он смог получить по сети и впоследствии раскрыл (например, дипломатическую переписку), явно выходили за рамки того, что ему и его сослуживцам необходимо было знать.

Принцип непрерывного обучения правилам безопасности. Знакомство с политикой безопасности предприятия при поступлении на работу должно дополняться регулярными разъяснениями всех вносимых в нее изменений; необходимо донести до сотрудников важность обеспечения безопасности и объяснить, что в связи с этим ожидается от них.

Средства безопасности технического уровня

Именно этот вид средств защиты в основном рассматривается в данной книге. Технические средства и методы можно разделить на программные, аппаратные и программно-аппаратные. Программные средства включают защитные инструменты операционных систем (подсистемы аутентификации и авторизации пользователей, средства управления доступом, аудит и др.) и прикладные программы, предназначенные для решения задач безопасности (системы обнаружения и предотвращения вторжений, антивирусные средства, прокси-серверы). Примером аппаратных средств, специализирующихся на информационной защите, являются источники бесперебойного питания, генераторы напряжения, средства контроля доступа в помещения и др. К аппаратно-программным средствам относятся, например, некоторые анализаторы сетевого трафика и межсетевые экраны. И хотя данный уровень средств называется техническим, к нему также относят математические методы (методы криптографии), алгоритмы (эвристический алгоритм расчета времени оборота в протоколе TCP), абстрактные модели (модели контроля доступа) и т. п.

Принципы защиты информационной системы

Далее рассмотрены принципы построения системы обеспечения информационной безопасности, многие из которых имеют универсальный характер и применимы для защиты систем самой разной природы.

Подход сверху вниз

Проектирование системы защиты должно идти сверху вниз.

Подход «сверху вниз», где понятие «верх» означает руководство предприятия, а «низ» — уровень рядовых сотрудников, соответствует универсальному принципу движения от

общего к частному. При таком подходе все принципиальные решения принимаются топ-менеджментом, затем руководители промежуточных уровней преобразуют их в более развернутые планы и частные решения, которые, наконец, доводятся в виде инструкций и в разной степени формализованных процедур до уровня исполнителей.

Именно руководители предприятия определяют стратегически важные объекты защиты, они оценивают риски, которые может понести предприятие в результате разрушения тех или иных информационных активов, они намечают стратегию защиты информационных ресурсов.

Такой подход является эффективным, так как, во-первых, руководители хорошо знают бизнес предприятия и могут правильно оценить риски, а следовательно, определить, какие именно информационные ресурсы нужно защищать особенно тщательно, а какие могут быть оставлены «без присмотра». Во-вторых, эти люди непосредственно несут ответственность за то, насколько эффективной окажется проектируемая система, а также обладают полномочиями для принятия критически важных решений.

Противоположный подход — «от частного к общему», или «снизу вверх», успешно используемый в некоторых сферах деятельности (например, в научных исследованиях), — совершенно неприменим для проектирования сложных технических систем, к которым относится система обеспечения безопасности. Решения, принимаемые на уровне специалистов отдельных подразделений, могут оказаться несогласованными и не способствовать достижению глобальной цели. Известен пример, когда системный администратор на свой страх и риск, приложив большие усилия и потратив значительные средства, обеспечил надежную защиту базы данных, а в ней, как впоследствии выяснилось, хранилась легко восстанавливаемая информация, которая не представляла для бизнеса особой ценности. Или другой пример. Руководитель отдела сетевой безопасности предприятия-провайдера Интернета для защиты транспортной подсистемы от внешних атак приобрел на основании собственного решения некое средство анализа сетевого трафика, способное собирать метаданные[1] о трафике с помощью установленных в сети маршрутизаторов, в том числе IP-адреса отправителей и получателей каждого сеанса некоторого клиента с некоторым сервером. Однако после того как система была внедрена, оказалось, что метаданные, собираемые этим сетевым анализатором, не обеспечивают анонимности, так как позволяют сопоставить имена пользователей и их IP-адреса, что противоречит федеральному закону «Об информации, информационных технологиях и о защите информации», который гласит, что в области защиты информации должны соблюдаться «неприкосновенность частной жизни, недопустимость сбора, хранения, использования и распространения информации о частной жизни лица без его согласия». Если бы решение о выборе сетевого анализатора принималось в рамках нисходящего процесса проектирования, то законодательное ограничение о персональных данных, зафиксированное на самом верхнем уровне в политике безопасности, последовательно передаваясь вниз по «лестнице» принятия решений, обязательно бы достигло уровня отдела сетевой безопасности и было бы учтено при выборе продукта.

---

[1] Метаданные — это служебная информация, используемая при проведении сеанса связи. Например, для электронной почты метаданными являются электронные адреса и IP-адреса отправителя и получателя, название темы письма, дата, время отправления и получения и др., то есть все, кроме содержания письма.

Защита как процесс

Защита должна представлять собой непрерывный, циклический, проактивный процесс.

Задача информационной защиты не может быть решена раз и навсегда, напротив, работа по защите ИС должна идти *непрерывно* на протяжении всего существования защищаемой системы.

Все системы безопасности уникальны, поскольку отражают специфику конкретных предприятий, для защиты которых они предназначены. Но какие бы различные цели ни преследовались при их создании, какие бы различные технологии ни использовались, какие бы индивидуальные решения ни принимались, общий ход проектирования для всех правильно построенных систем защиты должен иметь *циклический* характер, поскольку основой процесса обеспечения безопасности является следующая повторяющаяся последовательность:

1. Анализ состояния защищаемой системы, ее уязвимостей и угроз.
2. Оценка рисков и управление рисками.
3. Разработка политики всех уровней.
4. Реализация принятых решений, направленных на снижение рисков.
5. Возвращение к пункту 1.

Процесс обеспечения безопасности по возможности должен иметь *проактивный* (упреждающий), а не реактивный характер. При реактивном подходе защита заключается в принятии мер уже после того, когда нарушение безопасности произошло. Очевидно, что для успешности отражения атаки в первую очередь важны правильно выбранные действия и скорость их выполнения, а как раз этого трудно ожидать в ситуации кризиса. Поэтому более предпочтительным является проактивный подход, когда для защиты от вероятных угроз в спокойной обстановке проводится основательная подготовка оборонительных мер: устанавливаются необходимые технические средства, продумываются действия персонала, составляются и документируются инструкции — то есть делается все, что только может быть сделано заранее.

Эшелонированная защита

Эффективная защита обеспечивается путем многократного резервирования средств безопасности.

Надежность решения любой задачи повышается, если использовать резервирование. Задача обеспечения безопасности не является здесь исключением. Так, например, для обеспечения физической сохранности важного документа могут применяться самые разные средства защиты: дверные замки, датчики разбития окон, противопожарные сигнальные устройства, тревожная кнопка, сейф и масса других полезных приспособлений.

Информационная система существует в окружении гораздо более изощренных и многообразных угроз, здесь тем более невозможно найти панацею — одно-единственное средство, которое могло бы со стопроцентной надежностью противостоять всем видам атак. Поэтому на пути к защищаемому информационному ресурсу, как правило, устанавливают несколько барьеров. Вместе с тем возникает резонный вопрос: если ни одно из средств обеспечения безопасности не является абсолютно надежным и в принципе может быть преодолено злоумышленником, то в чем смысл нескольких защитных рубежей? Ответ состоит в том,

что многократное резервирование в системах защиты служит не столько для того, чтобы какое-то из защитных средств продублировало отказавшее, а главным образом для того, чтобы заставить преступника *потратить как можно больше времени* на преодоление череды защитных барьеров. Замедление атаки повышает шанс ее обнаружения и принятия адекватных мер.

Рассмотрим, например, как реализуется принцип эшелонированной защиты в случае, когда необходимо обеспечить безопасность данных, хранящихся на одном из хостов внутренней локальной сети предприятия. На рис. 26.12 концентрическими окружностями представлены рубежи обороны, каждый из которых добавляет к уже накопленному защитному потенциалу собственные средства защиты (некоторые виды этих средств обеспечения безопасности рассмотрены в следующих главах).

Рис. 26.12. Рубежи обороны ИТ-системы

Самый внешний слой (организационно-административный) решает задачу безопасности данных, затрудняя злоумышленникам физический доступ к данным, с этой целью разрабатываются и применяются административные и организационные меры безопасности, такие как проверка персонала при приеме на работу, взаимный контроль персонала, ограничение использования переносных портативных носителей и др.

Следующий слой также направлен на защиту от физического проникновения, но другими средствами — средствами физической защиты: ограждения, освещение, видеокамеры, контроль входа в здание, двери с кодовыми замками и т. п.

Далее вступают в действие технические средства безопасности, которые для сети с типовой структурой включают следующие рубежи защиты:

- внешняя сеть — для защиты от проникновения применяются средства регистрации входа, аудит, защитные свойства VPN;
- периметр внутренней сети — защита усиливается за счет файервола и прокси-серверов;
- внутренняя сеть — добавляются системы обнаружения и предотвращения вторжений сетевого уровня;
- хост — дополнительно проводятся процедуры аутентификации и авторизации, работают программный файервол, антивирус, системы обнаружения и предотвращения вторжений уровня хоста;
- данные — механизм контроля доступа и шифрование.

Сбалансированная защита

Степень защищенности системы измеряется защищенностью ее самого слабого звена.

Этот принцип можно сформулировать и несколько по-другому: при построении системы безопасности необходимо обеспечить баланс стойкости всех ее компонентов. Например, если в сети все сообщения шифруются, но ключи легкодоступны, то эффект от шифрования нулевой.

Из данного принципа можно сделать также следующее заключение: если у злоумышленника существует несколько путей нанести урон системе и один из этих путей имеет слабую защиту, то нет смысла добиваться высокого качества защиты других путей. То есть если внешний трафик сети, подключенной к Интернету, проходит через мощный сетевой экран, но пользователи имеют возможность связываться с узлами Интернета по коммутируемым линиям через локально установленные модемы, то деньги (как правило, немалые), потраченные на сетевой экран, можно считать выброшенными на ветер. В таких случаях оказывается полезным еще один принцип — *принцип единого контрольно-пропускного пункта*, который заключается в том, что весь входящий во внутреннюю сеть и выходящий во внешнюю сеть трафик проходит через единственный узел сети, например межсетевой экран.

Необходимость баланса стойкости разных компонентов системы безопасности особенно ярко иллюстрируется провалами силовых ведомств, когда мощные организации, располагающие гигантскими ресурсами защиты, допускают существование явных прорех в своих системах безопасности. Так, известен случай, когда дисковый накопитель с классифицированными данными вооруженных сил США был случайно обнаружен продающимся на базаре в Ираке. Причина — использование ненадежных процедур утилизации данных и аппаратуры, хотя для остальных стадий существования данных — хранения и передачи — использовались мощные алгоритмы шифрования. Еще два примера связаны с масштабными утечками сверхсекретных данных — Мэннинг (2010) и Сноуден (2013). В обоих случаях очевидным слабым звеном стало управление персоналом: несмотря на то что незадолго до инцидентов в поведении потенциальных нарушителей их коллегами отмечались «странности», а также то, что в карьере каждого из них произошли события, которые обычно квалифицируются как провоцирующие факторы, в отношении них не было предпринято никаких расследований. Кроме того, в обоих случаях не сработал контроль использования портативных запоминающих устройств. В результате за пределы зоны безопасности были вынесены сотни тысяч секретных файлов, записанные Мэннингом на компакт-дисках и Сноуденом на устройствах флеш-памяти.

Компромиссы системы безопасности

Система обеспечения безопасности создается в результате компромисса между качеством защиты, с одной стороны, и затратами на разработку этой системы — с другой. Под качеством здесь понимается комплекс характеристик — функциональное разнообразие, надежность защиты, удобство работы сотрудников, поддерживающих систему безопасности, сотрудников других подразделений предприятия. Поясним природу необходимых компромиссов, используя очень упрощенную модель, описывающую финансовую сторону внедрения системы безопасности.

Назначение системы безопасности — сократить прогнозируемый совокупный ущерб, который мог бы быть нанесен предприятию, если бы система защиты отсутствовала. Пусть в исходном состоянии, до внедрения системы защиты, ущерб предприятия от атак (риск) оценивался как *L_before*.

При внедрении системы защиты возможный ущерб от атак снизился и оценивается как *L_after*, однако в позиции «убытки» у предприятия добавились затраты *N* на внедрение системы безопасности. Кроме того, к убыткам предприятия должны быть отнесены те потери *U*, которые предприятие понесло из-за снижения производительности в результате внедрения системы безопасности. (Такое снижение может быть вызвано как дополнительными затратами вычислительных ресурсов, так и необходимостью выполнения сотрудниками предприятия дополнительных процедур, связанных с безопасностью.)

Очевидно, что решение о внедрении системы безопасности можно считать экономически обоснованным только в том случае, если потери от риска *L_after* в совокупности с затратами на систему безопасности *N* и потерями из-за снижения производительности *U* окажутся меньше исходного значения ущерба от риска *L_before*:

$$N + U + L\_after < L\_before. \quad (1)$$

Из этого соотношения следует, что надо стремиться уменьшать каждое из слагаемых в левой части неравенства. Проблема, однако, состоит в том, что они не являются независимыми и уменьшение одного из них может вызвать увеличение других. Так, например, затраты на систему безопасности положительно влияют на защищенность предприятия, то есть чем больше *N*, тем меньше *L_after* (что, к сожалению, не всегда справедливо в реальных разработках). Значит, при создании системы безопасности необходимо стремиться к некоторому компромиссному варианту, который минимизирует выражение в левой части неравенства.

> *При создании системы безопасности необходим компромисс между затратами и рисками.*

В соответствии с нашей идеальной моделью слагаемые *L_after* и *N* действуют разнонаправленно, а значит, вложение денег в защиту выгодно, только если ущерб *L_after* снижается быстрее, чем растут затраты *N*. Реальность, однако, намного сложнее, и рекомендация ориентироваться на соотношение скоростей является сугубо абстрактной. В то же время в практической деятельности можно использовать тот факт, что возможный ущерб *L_after* никогда не снижается ниже некоторого порога.

Действительно, создание абсолютно непроницаемой защиты невозможно, так как у атакующих всегда остается теоретическая возможность взломать любую защиту, это вопрос только времени и тех средств, которыми располагают злоумышленники, а значит, рано или поздно наступает такой момент, когда становится бессмысленным продолжать вкладывать деньги в систему безопасности. Остается вопрос: на каком уровне затрат надо остановить-

ся[1]? В первом грубом приближении ответ следует из неравенства (1), которое выражает условие экономической обоснованности затрат на систему защиты. Поскольку слагаемые U и L_after — положительные числа, то необходимым, но не достаточным условием справедливости неравенства является отношение:

$$N < L\_before.$$

Этот результат часто формулируют в форме *принципа разумной достаточности*:

Затраты на обеспечение безопасности информации должны быть по крайней мере не больше, чем величина потенциального ущерба от ее утраты.

Вопрос «Когда следует остановиться?» можно сформулировать по-другому: какой уровень защищенности системы является достаточным? Для ответа на этот вопрос разработчикам системы безопасности предлагается встать на место злоумышленника и попытаться оценить, какой уровень защиты злоумышленник мог бы посчитать неприемлемым для себя. Так, например, вряд ли имеет смысл браться за добычу конфиденциальных данных, если эта работа настолько длительная, что к тому времени, когда секретная информация попадет в руки, она уже устареет и не будет представлять никакой ценности. Аналогично, никто (из экономически мотивируемых преступников) не будет заниматься взломом системы, если выгоды от обладания защищаемым ресурсом меньше, чем средства, потраченные на проведение атаки. Исходя из этих соображений формулируются еще два варианта принципа разумной достаточности, относящиеся к уровню защищенности, обеспечиваемому системой безопасности (стойкости):

- Стойкость системы безопасности считается достаточной, если время преодоления защиты превосходит время старения информации.
- Стойкость системы безопасности считается достаточной, если стоимость ее преодоления злоумышленниками превосходит стоимость полученной ими выгоды.

Таким образом, проектирование системы безопасности требует нахождения множества компромиссов между возможными затратами и возможными рисками (рис. 26.13). Так, в некоторых случаях можно отказаться от дорогостоящего файервола в пользу стандартных средств фильтрации обычного маршрутизатора, в других же приходится идти на беспрецедентные затраты.

Рис. 26.13. Компромиссы системы безопасности

[1] Здесь идет речь только об остановке в пределах данного цикла создания системы безопасности, на новом витке развития системы могут быть приняты другие решения.

При создании системы безопасности необходим компромисс между ее эффективностью и эффективностью защищаемого бизнеса.

При внедрении любой системы защиты производительность предприятия может только уменьшаться, так как, во-первых, дополнительные задачи, связанные с СОИБ, обременяют ИТ-инфраструктуру: часть вычислительных ресурсов — процессорное время, память, пропускная способность линий связи, затраты на обслуживание и администрирование и др. — идет на решение задач безопасности; во-вторых, сотрудникам предприятия в связи с внедрением (или усовершенствованием) системы обеспечения безопасности приходится выполнять дополнительные требования: например, блокировать компьютер при каждой отлучке от рабочего места, проходить дополнительно контроль на входе путем анализа сетчатки глаза, получать разрешение на использование ресурсов Интернета и т. п. Такая дополнительная работа доставляет сотрудникам неудобства и приводит в конечном счете к снижению производительности компании. В такой ситуации необходимо провести анализ, на основании которого должен быть найден компромисс между качеством системы безопасности и ее влиянием на удобство работы персонала и производительность бизнеса.

Следует заметить, что существуют примеры, когда внедрение системы безопасности благоприятно сказывалось на удобстве работы сотрудников и производительности основного бизнеса. Это, однако, не противоречит всему сказанному: так случается, когда в рамках системы безопасности внедряется технология *многоцелевого* назначения. Например, технология виртуальных частных сетей на основе MPLS предоставляет помимо защищенности ряд других преимуществ, в том числе возможность организации удобного и производительного удаленного доступа к серверам корпоративной сети.

В большинстве же случаев процедуры обеспечения безопасности только затрудняют работу. Поэтому так важно проводить разъяснительную работу, убеждать работников предприятия в необходимости следования правилам и процедурам безопасности.

Шифрование — базовая технология безопасности

Основные понятия и определения

Шифрование является краеугольным камнем всех служб информационной безопасности, будь то система аутентификации или авторизации, защищенный канал или средства безопасного хранения данных. Прежде чем перейти к конкретным методам и алгоритмам шифрования, давайте определим некоторые базовые понятия криптографии.

Шифрование — это обратимое преобразование информации в целях обеспечения конфиденциальности данных. **Дешифрование** — процедура, которая, будучи примененной к зашифрованному тексту[1], снова приводит его в исходное состояние.

Пара процедур — шифрование и дешифрование — называется **криптосистемой**. Обычно криптосистема предусматривает наличие специального элемента — **секретного ключа**.

---

[1] Информацию, над которой выполняются функции шифрования и дешифрования, мы будем условно называть «текстом», учитывая, что это может быть также числовой массив или графические данные.

В качестве ключа может выступать некоторый предмет, например книга, число или рисунок. Простейший метод шифрования — замена букв в шифруемом тексте в соответствии с тем или иным правилом. Например, каждой букве алфавита может ставиться в соответствие другая буква этого алфавита, сдвинутая на некоторое число позиций влево или вправо. В качестве секретного ключа здесь выступает число, определяющее сдвиг.

Криптосистема считается *раскрытой*, если найдена процедура, позволяющая подобрать ключ за реальное время. Методы раскрытия криптосистемы, процедуры выявления уязвимости криптографических алгоритмов, выяснение секретного ключа называют **криптоанализом**, или взломом шифра. Попытку раскрытия конкретного шифра с применением методов криптоанализа называют **криптографической атакой**.

Например, классическим методом криптоанализа, применяемым для раскрытия шифров, основанных на перестановке или замене букв, является частотный анализ. Для текстов, написанных на определенном языке, относящихся к определенной сфере знаний, существуют устойчивые статистические данные о частоте, с которой встречается в тексте та или иная буква или последовательность букв, включая некоторые слова. Обладая такими данными и проведя статистический анализ зашифрованного текста, можно выполнить обратную замену символов.

Сложность алгоритма раскрытия является одной из важных характеристик криптосистемы и называется **криптостойкостью**. В криптографии принято **правило Керкгоффса**, заключающееся в том, что *стойкость шифра должна определяться только секретностью ключа*. Так, все стандартные алгоритмы шифрования (например, AES, DES, PGP) широко известны[1], их детальное описание содержится в легкодоступных документах, но от этого их эффективность не снижается. Система остается защищенной, даже если злоумышленнику известно все об алгоритме шифрования, но он не знает секретный ключ.

Существует два класса криптосистем — **симметричные** и **асимметричные**. В симметричных схемах шифрования (классическая криптография) секретный ключ шифрования совпадает с секретным ключом дешифрования. В асимметричных схемах шифрования (криптография с открытым ключом) ключ шифрования не совпадает с ключом дешифрования.

Симметричное шифрование

На рис. 26.14 приведена модель симметричной криптосистемы. В данной модели три участника: два абонента, желающих, обмениваться шифрованными сообщениями, и злоумышленник, который хочет перехватить и каким-либо образом расшифровать передаваемые сообщения.

ПРИМЕЧАНИЕ

При объяснении алгоритмов шифрования здесь и далее мы будем называть участников обмена Алисой и Бобом, а злоумышленника, старающегося перехватить их сообщения, — Евой. Эти имена традиционно используются в криптографии.

[1] Вместе с тем существует немало фирменных алгоритмов, описание которых не публикуется для того, чтобы усилить защиту.

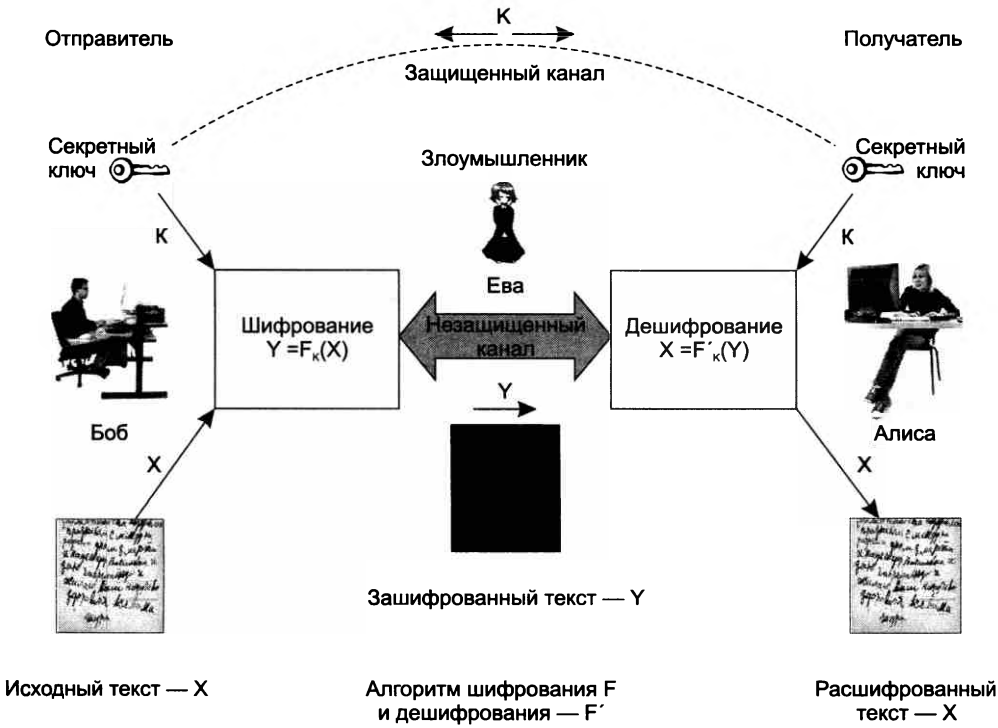


Рис. 26.14. Модель симметричного шифрования

В распоряжении Алисы и Боба имеется незащищенный канал передачи сообщений, который в принципе может прослушиваться злоумышленником. Поэтому они договариваются использовать шифрование, и для этого им нужен секретный ключ, известный только им двоим. Этот ключ им был передан (или один из них послал его другому) заранее по другому каналу — надежному. Боб и Алиса, получив ключ, находятся в абсолютно равном (симметричном) положении, каждый из них может как посылать шифрованные сообщения, так и получать и расшифровывать их. Для определенности на рисунке показана схема передачи сообщений со стороны Боба.

Боб зашифровывает свое сообщение — открытый текст X — функцией шифрования F с секретным ключом k и передает в открытый канал результат — шифрованный текст Y. Алиса получает Y и передает его на вход функции дешифрования F', которая выполняет в обратном порядке все действия, выполненные ранее функцией F. Это может быть сделано, только если на вход функции F будет подано то же самое значение параметра — значение ключа k. Алиса имеет секретный ключ и поэтому получает расшифрованное значение. При необходимости передавать шифрованные сообщения Бобу Алиса должна действовать аналогичным образом.

До недавнего времени наиболее популярным стандартным симметричным алгоритмом шифрования данных был **DES** (Data Encryption Standard). Для шифрования используются циклическая последовательность операций над битами шифруемого текста — перестановки, подстановки, логические операции двоичной арифметики. Эти операции применяются блочно, размер блока равен 64 бита. Некоторые операции над блоками шифруемых данных

связаны со значениями секретного ключа, также имеющего длину 64 бита. Промежуточный результат шифрования складывается по модулю 2 (операция XOR) с преобразованной двоичной последовательностью ключа. Полный шифрованный текст получается слиянием результатов шифрования для всех блоков исходного текста.

Процедура дешифрования выполняется в обратном порядке. Поскольку собственно алгоритм DES не является секретом и широко доступен, в том числе доступны все таблицы, описывающие перестановки, то стойкость алгоритма (степень сложности дешифрования) определяется только сложностью подбора ключа, которая прямо зависит от его длины.

Для того чтобы повысить криптостойкость алгоритма DES, был разработан его усиленный вариант, называемый **«тройным алгоритмом DES»**, который включает троекратное шифрование с использованием двух разных ключей. При этом можно считать, что длина ключа увеличивается с 56 до 112 бит, а значит, криптостойкость алгоритма существенно повышается. Но за это приходится платить производительностью — тройной алгоритм DES требует в три раза больше времени на реализацию, чем «обычный».

В 2001 году был стандартизован симметричный алгоритм шифрования **AES** (Advanced Encryption Standard). AES обеспечивает лучшую защиту, так как использует 128-битные ключи (а также может работать со 192- и 256-битоными ключами) и имеет более высокую скорость работы, кодируя за один цикл 128-битный блок в отличие от 64-битного блока DES. В настоящее время, помимо AES, распространенным симметричным алгоритмом шифрования является алгоритм Blowfish.

Криптостойкость всех симметричных алгоритмов зависит от качества ключа, это предъявляет повышенные требования к службе генерации ключей, а также к надежности канала обмена секретными ключами между участниками секретных переговоров.

Проблема распределения ключей

Симметричный подход к шифрованию изначально несет в себе очевидную проблему, называемую проблемой **распределения ключей** (key distribution), которая состоит в следующем. Отправитель и получатель хотят обмениваться секретными сообщениями, но в их распоряжении имеется незащищенный открытый канал. Поэтому они вынуждены использовать шифрование, но чтобы послать зашифрованное сообщение, нужно предварительно обменяться секретной информацией о значении ключа. Однако секретный ключ нельзя передать по открытому каналу. Если его зашифровать другим ключом, то опять возникает проблема доставки второго ключа. Получается замкнутый круг.

Единственным по-настоящему надежным решением этой проблемы является передача ключа при личной встрече абонентов. Однако при активном обмене требуется часто менять ключи, чтобы не дать возможности криптоаналитику собрать большое количество шифрованного материала, — известно, что чем больше зашифрованных сообщений окажется в руках криптоаналитика, тем легче ему раскрыть криптосистему. Кроме того, если злоумышленник перехватывает и сохраняет сообщения, зашифрованные одним и тем же ключом, то при раскрытии данного ключа они *все* окажутся скомпрометированными. Следовательно, необходимы частые личные встречи абонентов для обмена ключами, что, во-первых, не всегда возможно, а во-вторых, вообще делает бессмысленным обмен данными по каналу связи — действительно, зачем шифровать данные, если их можно лично передать при встрече.

Менее надежным способом распределения ключей является использование курьеров или других вариантов защищенной доставки ключей, но это решение тоже имеет очевидные изъяны. Существуют и другие приемы, не решающие, но смягчающие проблему распределения ключей. Например, у абонента может быть несколько секретных ключей, которые он должен использовать по разному назначению. Один ключ выдается ему на долгий срок, этот ключ применяется только для шифрования (дешифрования) других ключей — кратковременных, каждый из которых действителен только на время одного сеанса связи. И хотя в этом случае все равно остается проблема доставки долговременного ключа, уже нет необходимости его частой смены, так как этот ключ используется относительно редко и шифрует небольшие порции данных — сеансовые ключи.

Несмотря на различные усовершенствования процедуры распределения ключей, они не могут полностью устранить коренной изъян симметричных методов — *необходимость доставки секретного ключа по незащищенному каналу*.

Если проблема с ключами возникает в системе с двумя абонентами, то она многократно усугубляется в системе с большим числом абонентов. Пусть, например, n абонентов хотят обмениваться секретными данными по принципу «каждый с каждым», в этом случае потребуется $n\ (n - 1)/2$ ключей, которые должны быть сгенерированы и распределены надежным образом. То есть *количество требуемых ключей пропорционально квадрату количества абонентов*, что при большом числе абонентов делает задачу чрезвычайно сложной. Но именно такая ситуация наблюдается во всех современных сетях связи — телефонных, радио и компьютерных. Все это сделало чрезвычайно актуальной проблему распределения ключей.

Метод Диффи—Хелмана передачи секретного ключа по незащищенному каналу

В середине 70-х годов американские ученые Мартин Хеллман и Уилтфилд Диффи нашли способ, с помощью которого абоненты могли безопасно обмениваться секретными ключами без передачи их по каналу связи. Особенность этого открытия состоит в том, что оно противоречит всем интуитивным представлениям человека, делает возможным то, что кажется «очевидно» невозможным.

Метод Диффи—Хеллмана основан на использовании свойств односторонних функций.

Односторонняя функция (one-way function) — это функция $y = F(x)$, которая легко вычисляется для любого входного значения x, но обратная задача — определение x по заданному значению функции y — решается очень трудно. Примером односторонней функции может служить простейшая функция двух аргументов $F(p, q) = pq$, представляющая собой произведение двух простых чисел p и q, она вычисляется сравнительно просто, даже если числа p и q очень большие. Но чрезвычайно сложно решить обратную задачу (называемую факторизацией) — по произведению подобрать исходные два простых числа. Другой пример — функция $Y(x) = D^x \bmod P$, которая при некоторых ограничениях на параметры D и P является односторонней, то есть, зная Y, а также параметры D и P, нельзя без экстраординарных вычислительных усилий найти аргумент x.

Итак, пусть Алиса и Боб решили обмениваться шифрованными сообщениями, но в их распоряжении имеется только незащищенный открытый канал связи, при этом никаких возможностей встретиться или передать секретный ключ через кого-нибудь другого у них

нет. В соответствии с алгоритмом Диффи—Хеллмана для успешного решения задачи Алиса и Боб должны выполнить следующие действия. Предварительно они *открыто* договариваются о том, что будут использовать одностороннюю функцию $Y = D^x \bmod P$. Затем они договариваются о значениях параметров D и P. Пусть, например, они договорились, что $D = 7$ и $P = 13$, то есть функция имеет вид $Y = 7^x \bmod 13$. Еще раз подчеркнем, что в соответствии с алгоритмом Диффи—Хеллмана вся эта информация не является секретной, и даже если переговоры будут подслушаны Евой, это не даст ей возможности прочитать сообщения Алисы и Боба. Дальнейшие действия участников обмена описываются в табл. 26.1.

Таблица 26.1. Действия Алисы и Боба в соответствии с алгоритмом Диффи—Хеллмана

	Действия Алисы		**Действия Боба**	
1	Алиса секретным образом выбирает произвольное число A (закрытый ключ Алисы)	Пусть, например, $A = 2$	Боб также секретно выбирает произвольное число B (закрытый ключ Боба)	Пусть, например, $B = 4$
2	Алиса вычисляет значение a односторонней функции Y, используя в качестве аргумента свое секретное число A: то есть $a = D^A \bmod P$ (открытый ключ Алисы)	$a = 7^2 \bmod 13 = 10$	Боб также вычисляет значение b односторонней функции Y, используя в качестве аргумента свое секретное число B: $b = D^B \bmod P$ (открытый ключ Боба)	$b = 7^4 \bmod 13 = 2401 \bmod 13 = 9$
3	Алиса посылает Бобу свой открытый ключ a		Боб посылает Алисе свой открытый ключ b	
4	Алиса, получив от Боба число b, вычисляет по формуле $K = b^A \bmod P$ (разделяемый секретный ключ)	$K = 9^2 \bmod 13 = 81 \bmod 13 = 3$	Боб, получив от Алисы число a, вычисляет по формуле $K = a^B \bmod P$ (разделяемый секретный ключ)	$K = 10^4 \bmod 13 = 10000 \bmod 13 = 3$
5	По правилам модульной арифметики $b^A \bmod P = (D^B \bmod P)A \bmod P = D^{BA} \bmod P$	$K = 3$	По правилам модульной арифметики $a^B \bmod P = (D^A \bmod P)\, B \bmod P = D^{AB} \bmod P$	$K = 3$

В результате описанной процедуры на шаге 4 Алиса и Боб получили одно и то же число 3! Математические преобразования показывают, что вычисления Алисы и Боба всегда будут давать одинаковые результаты. Полученные в результате числа они могут использовать в качестве известного только им ключа для различных симметричных методов шифрования.

Посмотрим, может ли Ева подобрать разделяемый секретный ключ Алисы и Боба. Пусть на шаге 3, когда Алиса и Боб посылали друг другу свои открытые ключи a (10) и b (9), Ева смогла перехватить эти числа (ведь канал является открытым) и теперь пытается вычислить разделяемый секретный ключ. Зная число a, которое Алиса послала Бобу, Ева хочет повторить действия Боба и вычислить разделяемый секретный ключ по формуле $10^B \bmod 13$. Для этого ей требуется закрытый ключ Боба B, который он, однако, хранит секретно от всех. Зато Ева знает, что Боб использовал свой закрытый ключ B, когда вычислял значение своего открытого ключа — b. То есть задача будет решена, если Ева сможет подобрать такое значение B, чтобы значение $7^B \bmod 13$ равнялось 9. Но именно это практически неразрешимо, поскольку функция $7^B \bmod 13$ является односторонней. Таким образом, Алиса и Боб действительно получили секретный ключ.

Для того чтобы усложнить решение обратной задачи, то есть восстановление закрытого ключа Алисы или Боба по открытому, на параметры алгоритма накладываются некоторые ограничения, в том числе следующие:

- все параметры D, P, A, B должны быть целыми положительными числами;
- A и B должны быть большими числами порядка 10^{100};
- P должно быть большим простым числом порядка 10^{300}, причем желательно, чтобы $(P-1)/2$ также было простым числом;
- число D не обязательно должно быть большим, обычно оно выбирается меньше десяти, $D < P$.

Хотя алгоритм Диффи—Хеллмана стал прорывом в области криптографии, в его исходном состоянии он представлял скорее теоретическую, нежели практическую ценность. Устранив препятствие в виде необходимости надежного закрытого канала для передачи ключа, этот метод не снял проблемы квадратичной зависимости числа ключей от числа абонентов. Решение пришло очень скоро: уже через год после появления алгоритма Диффи—Хеллмана была теоретически доказана возможность принципиально нового подхода к шифрованию — асимметричного шифрования, при использовании которого (помимо прочих преимуществ) кардинально упрощается задача распределения ключей.

Концепция асимметричного шифрования

До сравнительно недавнего времени понятие «симметричное шифрование» не существовало просто потому, что все методы, которые использовались человечеством на протяжении нескольких тысяч лет, по современной классификации могли быть отнесены к классу симметричных, других просто не было. Более того, все эти тысячи лет существовала твердая убежденность, что в принципе никогда не может быть иных схем, кроме симметричной, когда отправитель шифрует с помощью секретного ключа, получатель с помощью этого же ключа расшифровывает!

Революция свершилась в конце 60-х — середине 70-х, когда с разницей в несколько лет две группы ученых, одна из которых — уже знакомые нам Диффи и Хеллман, а другая — сотрудники секретной правительственной лаборатории Великобритании[1] Эллис, Кокс и Уильямсон, независимо друг от друга изобрели принципиально новый подход к шифрованию, открывающий глобальные перспективы в области современных коммуникаций. Предельно упрощая, этот подход можно описать фразой: «отправитель шифрует сообщение с помощью одного ключа, а получатель расшифровывает его с помощью другого ключа». Как видим, здесь на двух сторонах обменного канала используются разные ключи, то есть присутствует асимметрия, соответственно все методы, основанные на таком подходе, стали называть «асимметричными».

Это удивительно, что за несколько тысяч лет не было ни одной известной науке попытки изобретения асимметричного метода шифрования, и вдруг, практически одновременно, две независимые группы ученых совершают это открытие! Возможно, причина кроется в том, что к концу 60-х годов совпало два обстоятельства: во-первых, возникла острая

[1] Известно, что исторически первыми были британские криптографы, которые открыли асимметричное шифрование на 6 лет раньше, чем Диффи и Хеллман, однако до 1997 года они не смогли обнародовать свои результаты, так как их работа имела гриф секретности.

потребность в новом типе шифрования, во-вторых, появились технические возможности реализации этой идеи.

Потребность была продиктована зрелостью таких видов массовых коммуникаций, как телефон, радио, компьютерные сети, для которых, во-первых, особенно важна секретность ввиду слабой защищенности публичных средств связи, а во-вторых, неприемлемы ограничения традиционных методов шифрования, выражающиеся в необходимости обмена секретным ключом для каждой пары абонентов. К концу 60-х годов стали отчетливо вырисовываться перспективы использования Интернета как мировой сети связи, и одновременно с этим стало приходить осознание того, что глобальная публичная сеть может выполнить свою миссию только в том случае, если миллионам ее пользователей будет предоставлена возможность защищенного обмена сообщениями. Эти темы особенно волновали военных разных стран, которых очень привлекала возможность распределенного управления вооруженными силами, но пугала невозможность гарантировать секретность передаваемых директив. И если в недалеком прошлом проблема распределения секретных ключей хотя и существовала, но была преодолимой, то в новых условиях она стала принципиальным препятствием.

К этому времени созрели технические возможности реализации вычислительно емких алгоритмов шифрования, к которым могут быть отнесены асимметричные алгоритмы. Массовое распространение получили компьютеры, обладающие такой вычислительной мощностью, которой до сих пор могли похвастаться только уникальные модели суперкомпьютеров. Это сделало шифрование обыденной операцией, которая может быть выполнена на обычном персональном компьютере.

Вот на таком историческом фоне была предложена концепция асимметричной криптосистемы, называемой также **шифрованием с открытым ключом**.

На рис. 26.15 представлена модель асимметричной криптосистемы. Так же как и в модели симметричного шифрования (см. рис. 26.14), здесь показаны три участника: отправитель (Боб), получатель (Алиса) и злоумышленник (Ева). В отличие от симметричной схемы шифрования, в которой наличие разделяемого секретного ключа автоматически означает возможность двустороннего защищенного обмена, здесь существует отдельная процедура для передачи зашифрованных сообщений в каждую из сторон. На рисунке показан вариант, когда зашифрованные сообщения могут быть посланы только Бобом в сторону Алисы, но не наоборот.

Итак, Алиса пожелала, чтобы Боб посылал ей зашифрованные сообщения. Для этого она сгенерировала пару ключей: **открытый ключ** (public key) E и **закрытый ключ** (private key) D. Для шифрования текста служит открытый ключ, но расшифровать этот текст можно только с помощью закрытого ключа. Алиса не хочет, чтобы кто-либо читал ее почту, поэтому она сохраняет закрытый ключ D (часто называемый также личным ключом) в секрете. Открытый же ключ E Алиса свободно передает всем, от кого хочет получать зашифрованные сообщения. Открытый ключ не представляет никакого секрета, Алиса может поместить его на своей странице в социальной сети или обнародовать в рекламе на телевидении. Все, кто хотят посылать Алисе зашифрованные сообщения, используют один и тот же ключ E, но при этом никто из них не может прочитать сообщения друг друга.

1. Алиса передает Бобу свой открытый ключ E по незащищенному каналу в незашифрованном виде.

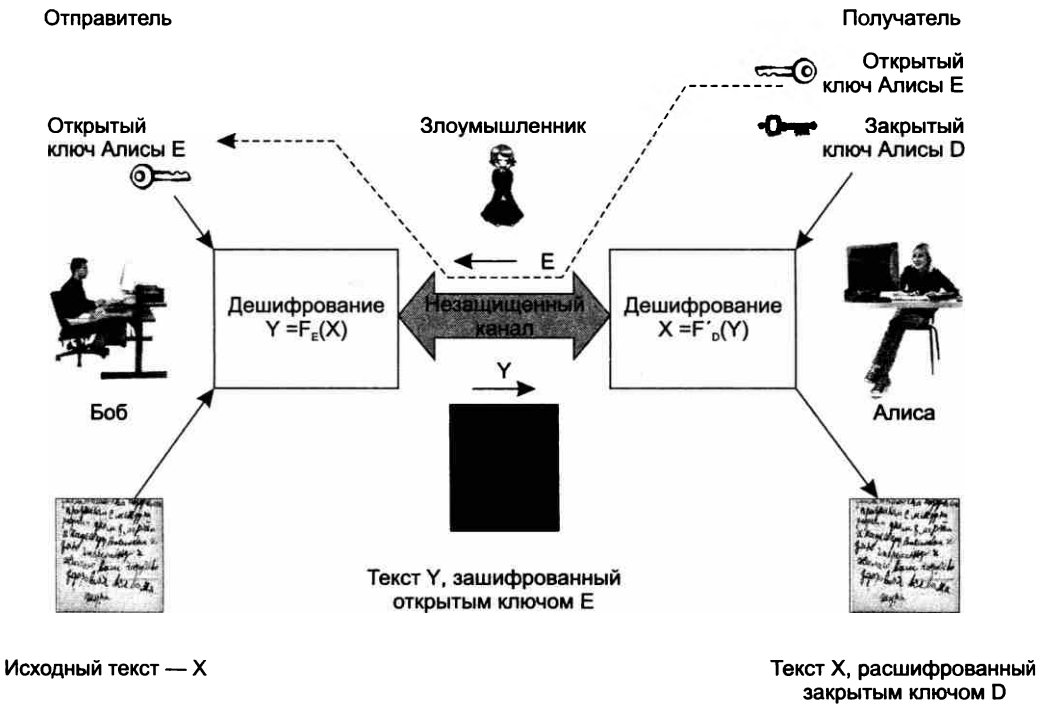


Рис. 26.15. Схема асимметричного шифрования

2. Боб шифрует свое сообщение X открытым ключом Алисы E и посылает зашифрованный текст $Y = F_E(X)$ по открытому каналу. Никто не может прочитать это сообщение. Даже сам Боб, если бы ему вдруг захотелось перечитать, что он там написал, не смог бы этого сделать, потому что для этого нужен закрытый ключ Алисы, которого у него нет.
3. Алиса получает шифрованное сообщение $Y = F_E(X)$ и расшифровывает его своим закрытым ключом D: $X = F'_D(Y)$.

Для того чтобы в сети все n абонентов имели возможность не только принимать зашифрованные сообщения, но и сами посылать таковые, каждый абонент должен обладать собственной парой ключей E и D. Всего в сети будет $2n$ ключей: n открытых ключей для шифрования и n секретных ключей для дешифрования. Таким образом решается проблема масштабируемости: *квадратичная зависимость количества ключей от числа абонентов в симметричных алгоритмах заменяется линейной зависимостью в асимметричных алгоритмах*. Решается и проблема доставки ключа, поскольку теперь он не является секретом, его можно без опаски передавать по открытому каналу. Злоумышленнику нет смысла стремиться завладеть открытым ключом, поскольку это не дает возможности расшифровать текст или вычислить закрытый ключ.

Алгоритм асимметричного шифрования RSA

Открыватели асимметричного подхода к шифрованию показали *концептуальную возможность* существования функций, позволяющих построить криптографическую систему,

в которой текст шифруется одним ключом, а расшифровывается — другим. Они также обрисовали те перспективы, которые открывает этот подход в деле решения проблемы распределения ключей. Ими были сформулированы два принципиальных требования, которым должны удовлетворять функции асимметричной криптосистемы:

- зашифрованное сообщение должно быть результатом вычислений односторонней функции, так чтобы никто не мог выполнить обратные преобразования и получить исходный текст;
- эта односторонняя функция должна быть сконструирована таким образом, чтобы у нее был некоторый секретный элемент, зная который получатель шифровки мог легко выполнить обратное преобразование.

Функции, которые удовлетворяют данным требованиям, назвали **односторонними функциями с потайным входом** (trapdoor function). Некоторое время ученым не удавалось найти функции, удовлетворяющие этим критериям, поэтому идея асимметричного шифрования не находила практического применения. Наконец, в 1978 году трое американских ученых, Ривест, Шамир и Адлеман, предложили долгожданный **алгоритм асимметричного шифрования RSA**, названный так по первым буквам их фамилий — Rivest, Shamir, Adleman. В табл. 26.2 описываются основные шаги алгоритма RSA.

Таблица 26.2. Последовательность действий участников обмена данными в соответствии с алгоритмом RSA

Действия Алисы и Боба	Числовой пример
Алиса произвольно выбирает два случайных простых числа P и Q. Они должны быть очень большими — от этого зависит стойкость алгоритма шифрования	В примере для простоты расчетов берутся очень маленькие числа. Пусть $P = 7$ и $Q = 13$
Алиса вычисляет два произведения $N = PQ$ $M = (P - 1)(Q - 1)$	$N = 91$ $M = 6 \times 12 = 72$
Алиса выбирает случайное целое число E, меньшее M и не имеющее с ним общих сомножителей	$E = 5$
Пара (E, N) — это открытый ключ Алисы, который она передает всем, от кого хочет получать шифрованные сообщения. Алиса посылает Бобу и всем остальным, с кем она желает вести защищенную переписку, свой открытый ключ (E, N)	(5, 91)
Алиса находит D такое, что $DE = 1 \bmod M$. Пара (D, N) — это закрытый ключ Алисы, который она не показывает никому. С этого момента она готова получать зашифрованные сообщения от Боба	$D \times 5 = 1 \bmod 72$ $D = 29$ (это число легко находится подбором, если учитывать признаки делимости на 5)
Боб получил открытый ключ Алисы и так же, как все остальные, имеющие доступ к этому ключу, может посылать Алисе зашифрованные сообщения. Он представляет свое сообщение в любом цифровом формате и разбивает его на блоки X таким образом, чтобы $0 < X < N$	Пусть секретный текст, посылаемый Бобом, состоит из одного символа R, который в коде ASCII имеет значение 1010010, или 82 в десятичном коде
Боб шифрует сообщение X открытым ключом (E, N): $C = X^E \bmod N$ и посылает Алисе зашифрованное сообщение C	$C = 82^5 \bmod 91 = \{82^3 \bmod 91 \times$ $\times 82^2 \bmod 91\} \bmod 91 = 10$ Вычисление модуля от степени числа упрощается при использовании следующего правила: $(Y^{a+b+c}) \bmod P = (Y^a \bmod P \times$ $\times Y^b \bmod P \times Y^c \bmod P) \bmod P$

Таблица 26.2 (*продолжение*)

Действия Алисы и Боба	Числовой пример
Алиса получает сообщение C и расшифровывает его своим закрытым ключом (D, N): $X = C^D \bmod N$	$X = 10^{29} \bmod 91 = \{10^1 \bmod 91 \times 10^4 \bmod 91 \times 10^6 \bmod 91 ...\} \bmod 91$ (внутри фигурной скобки четыре раза повторяется последний сомножитель — $10^6 \bmod 91$) $10 \bmod 91 = 10$; $10^4 \bmod 91 = 81$; $10^6 \bmod 91 = 1$ $X = \{10 \times 81 \times 1\} \bmod 91 = 82$
Результат расшифровки $X = 82$ совпадает с исходным секретным сообщением	

Еве для того, чтобы прочитать перехваченное сообщение C, требуется закрытый ключ Алисы (D, N). Но в ее распоряжении имеется только открытый ключ (E, N). Теоретически, зная открытый ключ, можно вычислить значение закрытого ключа. Однако необходимым промежуточным действием в этом преобразовании является нахождение простых чисел P и Q, для чего нужно разложить на простые множители очень большое число N, а это является чрезвычайно трудоемкой процедурой. Таким образом, здесь мы имеем дело с односторонней функцией $N = P \times Q$. Но для Алисы это же действие — разложение большого числа на два простых множителя — не представляет никакого труда, потому что она знает, как сконструировано это число N, она сама его вычислила, произвольно выбрав два сомножителя. Другими словами, Алисе известен «потайной вход» этой односторонней функции. Именно с огромной вычислительной сложностью разложения большого числа N на простые множители P и Q связана высокая криптостойкость алгоритма RSA.

Хотя информация об открытом ключе не является секретной, ее нужно защищать от подлогов, чтобы злоумышленник под именем легального пользователя не навязал свой открытый ключ, после чего с помощью своего закрытого ключа он мог бы расшифровывать все сообщения, посылаемые легальному пользователю, и отправлять свои сообщения от его имени. Решение проблемы дает технология **цифровых сертификатов**[1] — электронных документов, которые связывают конкретных пользователей с конкретными открытыми ключами.

Хеш-функции. Односторонние функции шифрования. Проверка целостности

В области информационной безопасности особое место занимает специальный класс односторонних функций, называемых хеш-функциями.

Хеш-функцией (hash function) называют такую одностороннюю функцию, которая, будучи примененной к некоторым данным, дает в результате значение, состоящее из фиксированного сравнительно небольшого и не зависящего от длины исходных данных числа байтов. Результат работы хеш-функции называют **хеш-кодом**, или **дайджестом**.

Рассмотрим, например, функцию взятия модуля $Y(x) = x \bmod n$, где операция $x \bmod n$ (то есть x по модулю n) дает в результате остаток от деления x на n. Эта функция, во-первых, является односторонней, так как, зная остаток от деления x на n, невозможно однозначно определить значение аргумента x, во-вторых, она относится к классу хеш-функций, по-

[1] См. раздел «Аутентификация на основе цифровых сертификатов» в главе 27.

скольку ее результат не зависит от аргумента x и всегда находится в диапазоне от 0 до $(n-1)$.

Хеш-функции называют также **односторонними функциям шифрования (ОФШ)**, где в качестве шифрованного представления исходных данных выступает дайджест. При этом знание дайджеста *не позволяет* и даже *не предполагает* восстановления исходных данных. Односторонние функции шифрования используют в разных целях, в том числе для обеспечения целостности и аутентичности информации. Пусть, например, требуется обеспечить целостность сообщения, передаваемого по сети. Отправитель и получатель договорились, что они будут использовать одностороннюю функцию H с секретным числом — ключом K — в качестве параметра. Прежде чем отправить сообщение X, отправитель вычисляет для него дайджест $M = H(X, K)$ и отправляет его вместе с сообщением X адресату (рис. 26.16). Адресат, получив данные X и M, применяет ту же самую ОФШ к переданному в открытом виде исходному сообщению X, используя известный ему секретный ключ K: $M' = H(X, K)$. Если значения дайджестов, вычисленного локально M' и полученного по сети M совпадают, значит, содержимое сообщения не было изменено во время передачи.

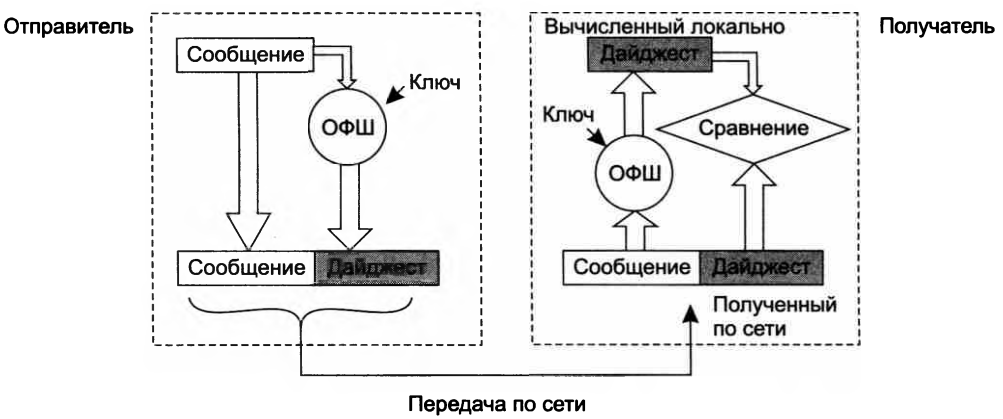


Рис. 26.16. Использование параметрической односторонней функции шифрования для контроля целостности

Хеш-функции широко используются в сетевых протоколах, в алгоритмах электронно-цифровой подписи, в механизмах аутентификации на основе паролей. Наиболее популярными в системах безопасности в настоящее время является серия хеш-функций MD2, MD4, MD5. Все они генерируют дайджесты фиксированной длины в 16 байт. Адаптированным вариантом MD4 является американский стандарт SHA, длина дайджеста в котором составляет 20 байт. Компания IBM поддерживает односторонние функции MDC2 и MDC4, основанные на алгоритме шифрования DES.

Выводы

Основными процедурами контроля доступа являются:

- идентификация — присвоение объектам и субъектам системы информационной системы уникальных имен — идентификаторов;

- аутентификация — процедура доказательства субъектом/объектом того, что он есть то, за что (кого) он себя выдает;
- авторизация — процедура контроля доступа субъектов (пользователей, вычислительных процессов, устройств) к объектам (например, файлам, приложениям, сервисам, устройствам) и предоставления каждому из них именно тех прав, которые им определены правилами доступа.

Понятие безопасной системы формализуется путем определения моделей безопасности. К числу наиболее распространенных можно отнести модели триады КЦД, гексады Паркера, STRIDE. В соответствии с моделью триады КЦД информационная система находится в состоянии безопасности, если она защищена от нарушений конфиденциальности, целостности и доступности.

К числу основных понятий управления рисками относятся:

- уязвимость — слабое звено информационной системы, которое, став известным злоумышленнику, может позволить ему нарушить ее безопасность;
- угроза — набор обстоятельств и действий, которые потенциально могут привести к нарушению безопасности системы (то есть к нарушению ее конфиденциальности, целостности и доступности, если пользоваться моделью КЦД);
- атака — реализованная угроза;
- ущерб — негативное влияние на систему, оказываемое проведенной атакой;
- риск — оценка ущерба от атаки с учетом ее вероятностной природы.

Успех в области информационной безопасности может принести только системный подход, при котором согласованно применяются средства защиты разных типов: законодательные, административные, процедурные и технические. Важным направлением законодательства, помимо преследования нарушителей в области безопасности, являются стандартизация и лицензирование. Основу административного уровня средств безопасности составляет политика безопасности, которая определяет стратегические направления информационной защиты предприятия. Средства безопасности процедурного уровня, основным инструментом которых является *человек*, решают задачи, поставленные вышележащим административным уровнем, с использованием технических средств, предоставляемых нижележащим техническим уровнем. Технический уровень включает подсистемы аутентификации, авторизации и аудита ОС, системы обнаружения и предотвращения вторжений, антивирусные средства, анализаторы сетевого трафика и файерволы, а также другие самые разнообразные программные и аппаратные средства; еще к нему относят математические методы, в частности методы криптографии.

Процесс построения системы защиты должен соответствовать следующим универсальным принципам:

- проектирование системы защиты должно идти сверху вниз;
- защита должна представлять собой непрерывный, циклический, проактивный процесс;
- эффективная защита обеспечивается путем резервирования средств безопасности;
- степень защищенности системы измеряется защищенностью ее самого слабого звена;
- при создании системы безопасности необходим компромисс между затратами и рисками.

Контрольные вопросы

1. Вставьте термины «уязвимость», «угроза» и «атака» в следующее предложение: «Необходимым условием осуществления... является наличие... и направленной на ее использование...».
2. Какие из перечисленных атак являются активными? Варианты ответов:
   а) прослушивание сетевого трафика;
   б) спуфинг;
   в) социальный инжиниринг;
   г) атака отказа в обслуживании;
   д) внедрение вируса.
3. Если система контроля доступа не разрешила пользователю распечатать документ на принтере, то можно сказать, что:
   а) имеет место отказ в обслуживании;
   б) пользователь не авторизован выполнять данную операцию;
   в) пользователь сделал попытку несанкционированного доступа к устройству;
   г) операция печати не входит в набор допустимых операций принтера.
4. Укажите правильные, на ваш взгляд, варианты продолжения тезиса «Политика информационной безопасности — это...». Варианты ответов:
   а) режим работы предприятия;
   б) общее руководство, определяющее главные направления деятельности в области защиты ИС;
   в) «закон», обязательный для исполнения всеми сотрудниками предприятия;
   г) совокупность документированных принципов, правил, процедур и подходов.
5. Какие из следующих утверждений являются ошибочными? Варианты ответов:
   а) при наличии достаточного бюджета предприятие может успешно завершить процедуру управления рисками и сконцентрироваться на других задачах;
   б) надежная система авторизации может компенсировать недостаточную стойкость паролей пользователей, которые они применяли при входе;
   в) затраты на обеспечение безопасности информации должны быть по крайней мере не выше, чем величина потенциального ущерба от ее утраты;
   г) всегда необходимо принимать меры к снижению риска.

ГЛАВА 27 Технологии аутентификации, авторизации и управления доступом

Технологии аутентификации

Как было определено в предыдущей главе, аутентификация применительно к вычислительной системе — это доказательство подлинности различных элементов этой системы при их взаимодействии. Пользователь при входе в систему должен предъявить системе доказательства, что он именно тот пользователь, идентификатор которого он вводит. Таким доказательством может служить пароль. Документ, полученный пользователем по электронной почте, должен сопровождаться дополнительной информацией, убеждающей пользователя, что документ не был изменен при передаче и что автором этого документа является именно тот человек, от имени которого это письмо было послано. Здесь доказательством может служить электронная подпись. Устройства, взаимодействующие по сети, должны доказать друг другу, что ни одно из них не подменено злоумышленником с целью ответвления или прослушивания трафика. Для этого в протоколе взаимодействия этих устройств должна быть предусмотрена процедура взаимной аутентификации. Взаимная аутентификация требуется и для организации безопасного сеанса пользователя и серверного приложения. Аутентификация может проводиться по отношению не только к отдельному пользователю, но и к группе пользователей. Методы аутентификации различаются в зависимости от того, что служит аутентификатором, а также от того, каким образом организован обмен аутентификационными данными между аутентифицируемым и аутентифицирующим элементами системы.

Факторы аутентификации человека

Абсолютно надежная аутентификация человека представляет собой теоретически неразрешимую задачу. Нет такого аутентификатора, который со стопроцентной надежностью доказывал бы аутентичность человека. Пароль можно перехватить, электронный ключ украсть, отпечаток пальца подделать, радужную оболочку глаза подменить качественным изображением. Более того, не существует научного доказательства невозможности совпадения у разных людей отпечатков пальцев или радужных оболочек глаза. Даже совпадение результатов анализа ДНК при современном уровне развития техники не может служить абсолютным доказательством аутентичности человека.

Однако на практике при аутентификации пользователей в вычислительных системах ограничиваются некоторым не стопроцентным, хотя и достаточно высоким уровнем достовер-

ности доказательства аутентичности человека. Аутентификаторы, которые используются при этом, разделяют на три класса:

- «что-то, что знаю» — к этому типу относятся многоразовые и одноразовые пароли, правила преобразования информации;
- «что-то, что имею» — различные миниатюрные устройства, называемые аппаратными аутентификаторами/ключами;
- «что-то, чем являюсь» — различные биометрические показатели аутентифицируемого.

Класс аутентификаторов называют *фактором*. Если в процедуре аутентификации предусматривается предъявление аутентифицируемым нескольких аутентификаторов, относящихся к разным классам, то такую аутентификацию называют многофакторной. Наибольшее распространение в настоящее время получила **двухфакторная аутентификация**, при которой пользователь предъявляет многоразовый пароль («что-то, что знаю») и аппаратный ключ («что-то, что имею»). Следует заметить, что в некоторых случаях термин «многофакторная аутентификация» служит для обозначения процедур **многоступенчатой аутентификации**, построенных на использовании нескольких аутентификаторов, относящихся к одному и тому же классу. Примером такой процедуры является аутентификация владельца банковского счета при его звонке в банк: сначала его просят назвать несколько букв из его пароля, а затем задают несколько вопросов с заранее согласованными и зафиксированными в базе данных аутентифицирующей организации ответами, например о памятном для него географическом пункте, девичьей фамилии матери и т. п.

Аутентификация на основе паролей

Пароль — это сохраняемая в секрете последовательность символов, либо выбранная пользователем, либо сгенерированная программным или аппаратным средством, либо назначенная администратором. Пароли относятся к аутентификаторам класса «что-то, что знаю».

Пароли бывают одноразовыми и многоразовыми. **Многоразовые пароли**, как это следует из их названия, могут использоваться для доказательства аутентичности многократно. В процедурах аутентификации, основанных на **одноразовых паролях**, аутентифицируемый должен каждый раз предъявлять новое значение пароля. Обычно для генерации одноразовых паролей применяются специальные программы или аппаратные устройства (см. далее).

Недостатки многоразовых паролей

Механизмы аутентификации на основе многоразовых паролей, обладая простотой и логической ясностью, традиционно являются самым популярным средством аутентификации. Однако им свойственны известные слабости. Это, во-первых, возможность раскрытия и разгадывания паролей, во-вторых, возможность «подслушивания» пароля при его передаче по сети путем анализа сетевого трафика. В-третьих, обладатели паролей могут стать жертвами социального инжиниринга. Так, например, беглый экс-сотрудник Агентства национальной безопасности США Эдвард Сноуден, работая системным администратором разведывательной базы США на Гавайях, использовал логины и пароли более 20 своих сослуживцев, чтобы получить доступ к секретным файлам. Он получал эти данные, объясняя, что они необходимы ему для работы.

Для снижения уровня угрозы раскрытия паролей администраторы сети, как правило, применяют встроенные программные средства, служащие для формирования *политики назначения и использования паролей*: задание максимального и минимального сроков действия пароля, хранение списка уже использованных паролей, управление поведением системы после нескольких неудачных попыток логического входа и т. п.

Многие пользователи пренебрегают угрозами, которые несут в себе легко угадываемые пароли. Так, червь Mumu, поразивший компьютерные сети в 2003 году, искал свои жертвы, подбирая пароли из очень короткого списка: password, passwd, admin, pass, 123, 1234, 12345, 123456 и пустая строка. Такая на удивление примитивная стратегия дала прекрасные (с точки зрения атакующей стороны) результаты — множество компьютеров было взломано.

В списке наиболее популярных паролей, применяемых пользователями Интернета при доступе к веб-серверам, опубликованном в августе 2013 года компанией Google, места в первой десятке занимают имена и даты рождения членов семьи и близких друзей, названия мест рождения, даты свадьбы, клички домашних животных, что-либо, связанное с любимой футбольной командой, и слово «password». Как видно из приведенного списка, для заинтересованного человека не составит большого труда подобрать эти пароли.

Но даже при выборе менее предсказуемого пароля вы все же рискуете, что он будет разгадан простым перебором всех возможных символов, — такой метод часто называют **брутфорс-атакой**[1]. Время подбора прямо зависит от разнообразия набора символов, из которого вы формируете свой пароль, и длины пароля. В табл. 27.1 приведены данные, характеризующие стойкость паролей, состоящих из 6 и 8 знаков, сформированных из разных наборов символов. Время определялось для специальной программы подбора паролей, выполняемой на компьютере со средними характеристиками[2]. Обратите внимание, насколько сильно возрастает время подбора пароля при увеличении его длины всего лишь на два знака. Так, при использовании только букв латинского алфавита (строчных и прописных) время подбора пароля из 8 знаков в 3000 раз больше, чем из 6 знаков!

Таблица 27.1. Сравнение стойкости паролей

Множество символов	Количество комбинаций		Время подбора пароля	
	6 знаков	8 знаков	6 знаков	8 знаков
Цифры от 1 до 9	1миллион комбинаций	100 миллионов комбинаций	Практически мгновенно	10 секунд
26 только прописных или только строчных букв латинского алфавита	309 миллионов комбинаций	200 миллиардов комбинаций	30 секунд	Менее 6 часов (в 720 раз дольше, чем для 6 знаков)
Сочетание 52 прописных и строчных букв латинского алфавита	19 миллиардов комбинаций	53 триллиона комбинаций	Полчаса	Два месяца (почти в 3000 раз дольше, чем для 6 знаков)
Прописные и строчные буквы, цифры и все символы (точка, двоеточие и т. п.)	782 миллиарда комбинаций	7,2 квадриллиона комбинаций	22 часа	57 лет (примерно в 22 700 раз дольше, чем для 6 знаков)

[1] Brute-force (англ.) — решать что-либо «в лоб», методом грубой силы.

[2] Данные взяты из статьи http://www.lockdown.co.uk/?pg=combi.

Серьезной проблемой использования многоразовых паролей является их ручная синхронизация. В обычной жизни нам требуется не один, а несколько паролей: для входа в компьютерную сеть предприятия, в котором мы работаем, для доступа к «личному кабинету» провайдера мобильной связи, для доступа к банковскому счету и еще для доступа к самым разным интернет-сайтам. Часто случается, что во всех этих случаях применяется один и тот же пароль (возможно, с небольшими вариациями), потому что у нас нет времени придумывать и, главное, запоминать новый пароль для доступа к новому ресурсу. Такое явление называют **ручной синхронизацией паролей**. Выполнив регистрацию на сайте, не заслуживающем доверия, вы сообщаете его владельцам свой пароль, который теперь может быть использован для доступа к другим вашим учетным данным, возможно, имеющим для вас критическое значение.

Слабостью паролей является также процедура реакции на неправильно введенный пароль. На первый взгляд естественным приемом, направленным на противодействие подбору паролей, кажется блокирование учетной записи, с которой было проведено некоторое количество (обычно три) неудачных попыток входа. Однако в некоторых ситуациях такой подход дает злоумышленнику прекрасную возможность быстро заблокировать работу предприятия. Действительно, идентификаторы пользователей являются менее защищенной информацией, чем пароли, к тому же они часто легко угадываемы (ADMIN, STUDENT, IVANOV, Natasha и т. п.) и их легче подсмотреть, так как они выводятся на экран. Поэтому злоумышленник может легко подобрать имена, выполнить по три неудачных попытки аутентификации для каждой учетной записи, вызвать их блокировку и привести таким образом систему в недоступное состояние. Снятие блокировок с учетных записей может стать серьезной проблемой, если таких записей очень много.

Наряду с паролями существует другой вариант использования аутентификаторов из класса «что-то, что знаю». Администратор заранее безопасным образом сообщает пользователю некоторое правило, например правило преобразования последовательности чисел в другие символы. Во время процедуры аутентификации система выводит на экран случайную последовательность чисел. Пользователь в соответствии с известным только ему и системе правилом преобразует их в другую последовательность символов, которую вводит в качестве пароля. Поскольку система также «знает» правило преобразования, она может проверить правильность введенного пароля. То есть изначально в данном случае в качестве разделяемого секрета выступает правило преобразования.

Строгая аутентификация в компьютерной сети на основе многоразовых паролей

Как правило, аутентификация пользователей в компьютерных сетях строится на основе централизованной схемы. На одном из серверов сети поддерживается база данных, в которой хранятся *учетные данные* обо всех пользователях сети. Учетные данные содержат наряду с другой информацией идентификаторы и пароли пользователей. Когда пользователь осуществляет логический вход в сеть, он набирает на клавиатуре своего компьютера свои идентификатор и пароль. По идентификатору пользователя в централизованной базе данных, хранящейся на сервере, находится соответствующая запись, из нее извлекается пароль и сравнивается с тем, который ввел пользователь. Если они совпадают, то аутентификация считается успешной, пользователь получает легальный статус и те права, которые определены для него системой авторизации.

Однако такая упрощенная схема имеет большой изъян: а именно — при передаче пароля с клиентского компьютера на сервер, выполняющий процедуру аутентификации, этот пароль может быть перехвачен злоумышленником. Поэтому в разных системах аутентификации применяются разные приемы, чтобы избежать передачи пароля по сети в незащищенном виде.

Аутентификация, в процессе которой используются методы шифрования, а аутентификатор не передается по сети, называется **строгой аутентификацией**.

Рассмотрим пример строгой аутентификации пользователей, реализуемой средствами ОС[1]. Пусть аутентификация пользователей сети выполняется на основе их паролей, хранящихся в зашифрованном виде в централизованной базе SAM (Security Accounts Manager). Пароли зашифровываются с помощью односторонней функции шифрования при занесении их в базу данных во время процедуры создания учетной записи для нового пользователя (рис. 27.1). Введем обозначение для этой односторонней функции — ОФШ1. Таким образом, пароль *P* хранится в базе данных SAM в виде дайджеста *d*(*P*). (Напомним, что знание дайджеста не позволяет восстановить исходный текст.)

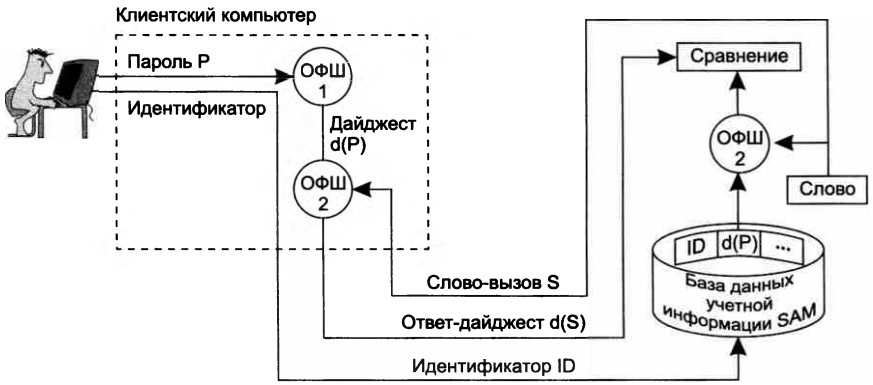


Рис. 27.1. Схема сетевой аутентификации на основе многоразового пароля

При логическом входе пользователь локально вводит в свой компьютер имя-идентификатор (ID) и пароль *P*. Клиентская часть подсистемы аутентификации, получив эти данные, передает запрос по сети на сервер, хранящий базу SAM. В этом запросе в открытом виде содержится идентификатор пользователя, но пароль в сеть ни в каком виде *не передается*.

К паролю на клиентской станции применяется та же односторонняя функция ОФШ1, которая была использована при записи пароля в базу данных SAM, то есть динамически вычисляется дайджест пароля *d*(*P*).

В ответ на поступивший запрос серверная часть службы аутентификации генерирует случайное число *S* случайной длины, называемое **словом-вызовом** (challenge). Это слово передается по сети с сервера на клиентскую станцию пользователя. К слову-вызову на клиентской стороне применяется односторонняя функция шифрования ОФШ2. В отличие от функции ОФШ1, функция ОФШ2 является параметрической и получает в качестве

[1] Аутентификация по данной схеме, наряду с другими методами, выполняется в ОС семейства Windows.

параметра дайджест пароля $d(P)$. Полученный в результате ответ $d(S)$ передается по сети на сервер базы SAM.

Параллельно этому на сервере слово-вызов S аналогично шифруется с помощью той же односторонней функции ОФШ2 и дайджеста пароля пользователя $d(P)$, извлеченного из базы SAM, а затем сравнивается с ответом, переданным клиентской станцией. При совпадении результатов считается, что аутентификация прошла успешно. Таким образом, аутентификация проходит без передачи пароля по каналам связи.

Заметим, что при каждом запросе на аутентификацию генерируется новое слово-вызов, так что перехват ответа $d(S)$ клиентского компьютера не может быть использован в ходе другой процедуры аутентификации.

Строгая аутентификация в протоколе CHAP

Другим примером строгой аутентификации может служить *аутентификация по квитированию вызова* (Challenge Handshake Authentication Protocol, CHAP), применяемая в протоколе PPP. Протокол PPP предусматривает два режима аутентификации: аутентификация по протоколу PAP, когда пароль передается по линии связи в открытом виде, и аутентификация по протоколу CHAP, при которой пароль по линии связи не передается и, следовательно, обеспечивается более высокий уровень безопасности.

Рассмотрим применение протокола CHAP при аутентификации удаленных пользователей, подключенных к Интернету по коммутируемому каналу. Здесь аутентифицирующей стороной является сервер провайдера, а аутентифицируемой — клиентский компьютер (рис. 27.2). При заключении договора клиент получает от провайдера пароль (пусть, например, это будет слово «parol»). Этот пароль хранится в базе данных провайдера в виде дайджеста $Z = d(\text{parol})$, полученного путем применения к паролю односторонней хеш-функции MD5.

В протоколе CHAP предусмотрено четыре типа сообщений: Success (успех), Challenge (вызов), Response (ответ), Failure (ошибка).

Аутентификация выполняется в следующей последовательности:

1. Пользователь-клиент активизирует некоторую программу удаленного доступа к серверу провайдера, вводя назначенные ему имя и пароль. Имя (на рисунке это «Moscow») передается по сети провайдеру в составе запроса на соединение, но пароль не передается в сеть ни в каком виде.
2. Сервер провайдера, получив запрос от клиента, генерирует псевдослучайное слово-вызов (пусть это будет слово «challenge») и передает его клиенту вместе со значением, идентифицирующим сообщение в рамках данного сеанса (ID), и собственным именем (здесь — «Paris»). Это сообщение типа Challenge. Для защиты от перехвата ответа аутентификатор должен использовать разные значения слова-вызова при каждой процедуре аутентификации.
3. Программа клиента, получив этот пакет, извлекает из него слово-вызов, добавляет к нему идентификатор и вычисленный локально дайджест $Z = d(\text{parol})$, а затем вычисляет с помощью все той же функции MD5 дайджест $Y = d\{(\text{ID}, \text{challenge}, d(\text{parol})\}$ от всех этих трех значений. Результат клиент посылает серверу провайдера в пакете Response.
4. Сервер провайдера сравнивает полученный по сети дайджест Y с тем значением, которое он получил, локально применив ту же хеш-функцию к набору аналогичных компонентов, хранящихся в его памяти.

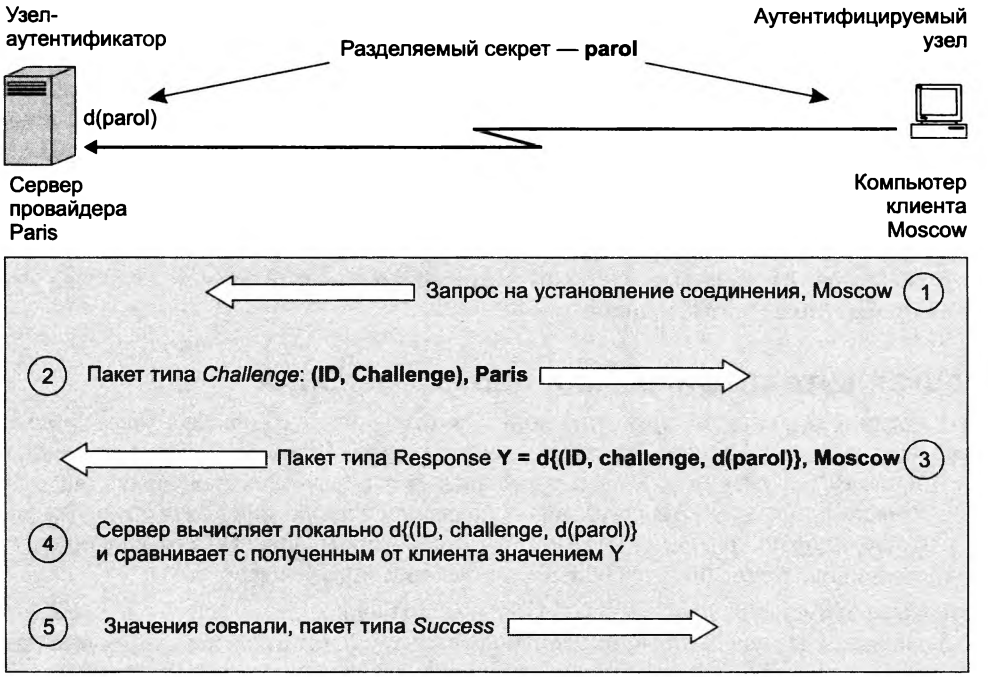


Рис. 27.2. Аутентификация по протоколу CHAP

5. Если результаты совпадают, то аутентификация считается успешной и аутентификатор посылает партнеру пакет Success.

Способ аутентификации, при котором многоразовые пароли пользователей хранятся в базе данных сервера в виде дайджестов, кажется вполне безопасным, ведь даже если злоумышленник сможет получить к ним доступ, он даже теоретически не сможет восстановить исходное значение паролей по дайджесту. Однако создатель первого червя Роберт Моррис решил эту проблему. Он разработал довольно простую программу, которая генерировала возможные варианты паролей, как используя слова из словаря, так и последовательным перебором символов. Для каждого сгенерированного слова вычислялся дайджест и сравнивался с дайджестами из файла паролей. Удивительно, но такая стратегия оказалась весьма эффективной, и хакеру удалось завладеть несколькими паролями.

Аутентификация на основе аппаратных аутентификаторов

Виды аппаратных аутентификаторов

Аппаратные аутентификаторы, называемые также аппаратными ключами, или токенами, относятся к разряду «что-то, что имею». Перечислим наиболее популярные из них (рис. 27.3):

- **Идентификаторы iButton** имеют ПЗУ, содержащее 64-битный код, однопроводной порт и схему, реализующая логику управления. При соприкосновении со считывателем

идентификатор iButton передает ему уникальный номер, считыватель проверяет его и инициирует сеанс обмена данными с идентификатором по определенному для этих двух устройств протоколу.

- **USB-ключи**, или **USB-токены**, подключаются непосредственно к USB-порту компьютера, исключая необходимость использования дорогостоящих считывающих устройств. Конструктивно USB-ключи выполняются подобно переносным устройствам памяти memory stick. Каждый USB-ключ имеет уникальный номер, присвоенный производителем.
- **Смарт-карты** (smart card — интеллектуальная карта) могут быть как контактными, так и бесконтактными. Контактные смарт-карты имеют на одной из сторон контактные площадки, через которые при соприкосновении с контактами считывателя происходит передача электропитания и аутентификационной информации. Считыватели имеют самое разнообразное конструктивное выполнение: например, они могут быть встроены в клавиатуру или в корпус компьютера. В бесконтактных смарт-картах предусматривается радиочастотный блок со встроенной антенной, проложенной по периметру карты. Антенна служит как для передачи аутентификационных данных, так и для извлечения энергии из электромагнитного поля, излучаемого считывателем. Смарт-карта содержит процессор, ПЗУ, в котором хранится криптографическая программа, ОЗУ, используемое как рабочая память, а также устройство EEPROM (электрически стираемое программируемое ПЗУ), содержащее изменяемые данные владельца карты. Аутентификация владельца карты осуществляется по уникальному серийному номеру карты, который присваивается ей на предприятии-изготовителе, а также по персональным данным владельца, хранящимся в памяти в зашифрованном виде.
- **Радиочастотные идентификаторы**, или **RFID-идентификаторы** (radio-frequency identification, RFID), изготавливаются в виде пластикового брелока, не имеющего никаких внешних информационных выходов-входов, а также никаких дисплеев и других видимых средств отображения информации. Внутри пластикового корпуса находится интегральная схема, которая хранит и обрабатывает информацию, а также миниатюрная антенна, предназначенная для передачи и приема радиосигналов. Система радиочастотной аутентификации помимо собственно RFID-идентификатора включает *считыватель радиочастотных сигналов*, который встраивается в электронные замки, вычислительные устройства и др. Обладатель RFID-идентификатора использует его как пропуск в помещения предприятия, поднося его к считывателю на стене перед дверью с электронным замком. Считыватель постоянно излучает радиочастотный сигнал, который при поднесении идентификатора на определенное расстояние (зависящее от типа устройства) принимается его антенной и передается в виде питания интегральной схеме. Эта энергия используется микросхемой для излучения аутентификационных данных в направлении считывателя. Этот же идентификатор может служить для доступа к принтеру и другим устройствам вычислительной системы. Кроме того, RFID-идентификатор позволяет службе безопасности отслеживать перемещение его владельца по всем помещениям, оснащенным соответствующими датчиками.

Аппаратные аутентификаторы, имеющие одинаковое конструктивное выполнение, могут реализовывать разные алгоритмы аутентификации. Например, смарт-карта может использоваться как генератор одноразовых паролей, как аутентификатор на основе закрытого ключа или в схеме аутентификации со словом-вызовом (все эти и другие методы аутентификации рассматриваются далее в этой главе).

Рис. 27.3. Электронные устройства-аутентификаторы

Общим недостатком всех аппаратных аутентификаторов является то, что они могут быть потеряны или, что значительно хуже, преднамеренно похищены. Любой человек, завладевший аутентификатором, теоретически получает в свое распоряжение все полномочия законного владельца.

Аутентификация на основе аппаратного генератора одноразовых паролей

Алгоритмы аутентификации, основанные на многоразовых паролях, не очень надежны. Пароли можно подсмотреть, разгадать или просто украсть. Более надежными оказываются схемы на основе программных или аппаратных **генераторов одноразовых паролей** (рис. 27.4).

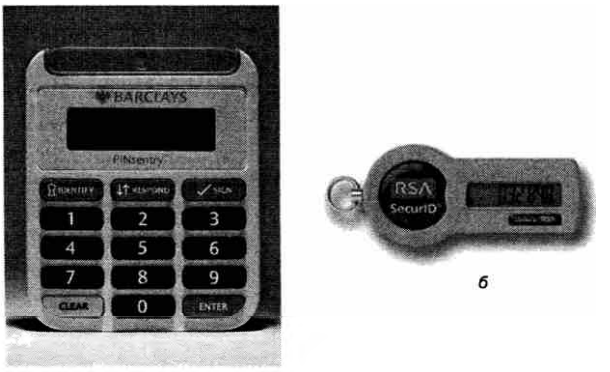


Рис. 27.4. Аппаратные ключи, генерирующие одноразовые пароли: *а* — ключ клиентов банка Barclays для доступа к своим счетам; *б* — аппаратный ключ компании SecurID

Независимо от того, какую реализацию системы аутентификации на основе одноразовых паролей выбирает пользователь, он, как и в системах аутентификации с применением многоразовых паролей, сообщает системе свой идентификатор, однако вместо того, чтобы вводить каждый раз один и тот же пароль, он указывает последовательность цифр, сообщаемую ему аппаратным или программным ключом. Через определенный небольшой период времени ключ генерирует другую последовательность — новый пароль. Сервер аутентификации проверяет введенную последовательность и разрешает пользователю осуществить логический вход. Сервер аутентификации может представлять собой отдельное устройство, выделенный компьютер или программу, выполняемую на обычном сервере.

Следует иметь в виду, что, как правило, системы аутентификации на основе одноразовых паролей рассчитаны на проверку удаленных, а не локальных пользователей.

Рассмотрим схему использования аппаратного генератора одноразовых паролей, в основе которой лежит **синхронизация по времени**. Этот популярный алгоритм аутентификации был разработан компанией Security Dynamics. Идея метода состоит в том, что аппаратный ключ и аутентифицирующий сервер по одному и тому же алгоритму вычисляют некоторое значение — одноразовый пароль. Алгоритм имеет два параметра:

- *разделяемый секретный ключ* представляет собой 64-разрядное число, уникально назначаемое каждому пользователю и хранящееся как в аппаратном ключе, так и в базе данных сервера аутентификации;
- *значение текущего времени.*

Если вычисленные значения совпадают, то аутентификация считается успешной.

Итак, пусть удаленный пользователь пытается совершить логический вход в систему с персонального компьютера (рис. 27.5). Аутентифицирующая программа предлагает ему ввести его личный персональный номер (PIN), состоящий из четырех десятичных цифр (на рисунке — 2360), а также одноразовый пароль — шесть цифр случайного числа, отображаемого в тот момент на дисплее аппаратного ключа (на рисунке — 112511). На основе PIN-кода сервер извлекает из базы данных информацию о пользователе, а именно его секретный ключ. Затем сервер выполняет вычисления по тому же алгоритму, который заложен в аппаратном ключе, используя в качестве параметров секретный ключ и значение текущего времени, проверяя, совпадает ли сгенерированное число с числом, введенным пользователем. Если они совпадают, то пользователю разрешается логический вход.

Потенциальной проблемой этой схемы является временная синхронизация сервера и аппаратного ключа. Ясно, что вопрос согласования часовых поясов решается просто. Гораздо сложнее обстоит дело с постепенным рассогласованием внутренних часов сервера и аппаратного ключа, тем более что потенциально аппаратный ключ может работать несколько лет. Компания Security Dynamics решает эту проблему двумя способами. Во-первых, при производстве аппаратного ключа измеряется отклонение частоты его таймера от номинала. Далее эта величина учитывается в виде параметра алгоритма сервера. Во-вторых, сервер отслеживает коды, генерируемые конкретным аппаратным ключом, и если таймер данного ключа постоянно спешит или отстает, то сервер динамически подстраивается под него.

Существует еще одна проблема, связанная со схемой временно́й синхронизации. Одноразовый пароль, генерируемый аппаратным ключом, действителен в течение некоторого интервала времени (от нескольких десятков секунд до нескольких десятков минут), то есть в течение этого времени одноразовый пароль, в сущности, является многоразовым. Поэтому теоретически возможно, что очень проворный хакер сможет перехватить PIN-код и одноразовый пароль с тем, чтобы также получить доступ в сеть в течение этого интервала.

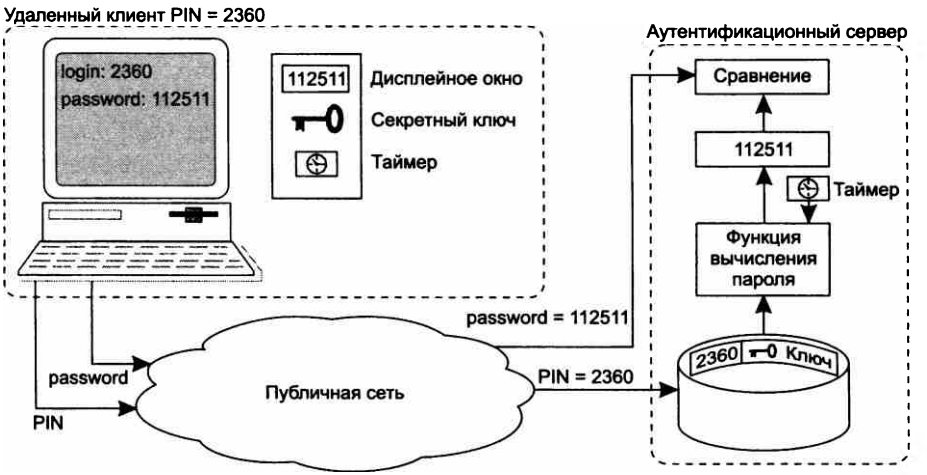


Рис. 27.5. Аутентификация на основе временно́й синхронизации

Схема временно́й синхронизации не требует наличия компьютера на стороне аутентифицируемого, для этих целей можно ограничиться простым терминалом или факсом. Пользователи могут даже вводить свой пароль с телефонной клавиатуры, когда звонят в сеть для получения голосовой почты.

Аутентификация по схеме «запрос-ответ»

Другая схема применения аппаратных ключей, называемая часто **запрос-ответ**, основана на идее, очень сходной с идеей строгой аутентификации, рассмотренной в предыдущем разделе. В том и в другом случае применяется слово-вызов. Когда пользователь пытается осуществить логический вход, аутентификационный сервер передает ему запрос в виде некоторого случайного числа (слово-вызов на рис. 27.6). Аппаратный ключ пользователя зашифровывает это случайное число (например, по алгоритму DES) и секретный ключ пользователя. Секретный ключ пользователя хранится в базе данных сервера и в памяти аппаратного ключа. В зашифрованном виде слово-вызов возвращается на сервер. Сервер, в свою очередь, также зашифровывает сгенерированное им самим случайное число с помощью того же алгоритма шифрования и того же секретного ключа пользователя, а затем сравнивает результат с числом, полученным от аппаратного ключа. Как и в методе временно́й синхронизации, в случае совпадения этих двух чисел пользователю разрешается вход в сеть.

Механизм со словом-вызовом имеет свои ограничения — он обычно требует наличия компьютера на каждом конце соединения, так как аппаратный ключ должен иметь возможность как получать, так и отправлять информацию. Схема «запрос-ответ» уступает схеме временно́й синхронизации по простоте использования. Для логического входа с помощью схемы временно́й синхронизации пользователю достаточно набрать 10 цифр. Схемы же «запрос-ответ» могут потребовать от пользователя выполнения большего числа ручных действий. В некоторых схемах «запрос-ответ» пользователь должен сам ввести секретный ключ, а затем набрать на клавиатуре компьютера полученное с помощью аппаратного ключа зашифрованное слово-вызов.

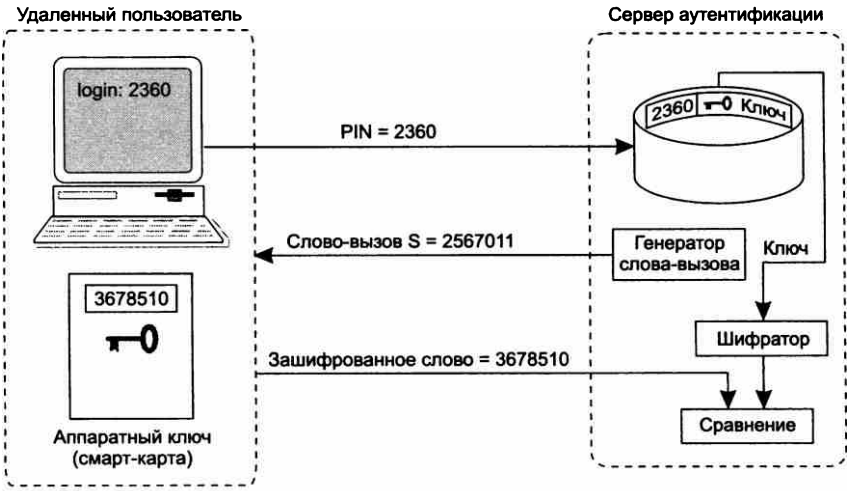


Рис. 27.6. Аутентификация по схеме «запрос-ответ»

Аутентификация информации. Электронная подпись

Аутентификация данных включает:

- подтверждение *целостности* хранящихся и переданных по сети данных и программ, то есть установление факта того, что они не подвергались модификации;
- доказательство *авторства* сообщения (документа, программы), в том числе и для недопущения отказа от авторства;
- доказательство *легальности* приобретения программного обеспечения.

Все эти задачи в той или иной мере могут быть решены посредством электронной подписи.

Согласно терминологии, утвержденной Международной организацией по стандартизации (ISO), под термином **электронная (цифровая) подпись** понимаются методы, позволяющие устанавливать подлинность автора сообщения (документа) при возникновении спора относительно авторства. Основная область применения цифровой подписи — финансовые документы, сопровождающие электронные сделки, документы, фиксирующие международные договоренности, и т. п.

Подчеркнем, что электронная подпись не ставит задачи обеспечения конфиденциальности сообщений.

Хотя для получения подписи могут использоваться симметричные алгоритмы, более распространенными являются алгоритмы на основе открытого и закрытого ключей. На рис. 27.7 показана схема формирования цифровой подписи по алгоритму RSA. Каждый пользователь сети имеет свой закрытый ключ (D, n), необходимый для формирования подписи, а соответствующий этому секретному ключу открытый ключ (E, n), предназначенный для проверки подписи, известен всем другим пользователям сети. Подписанное сообщение состоит из двух частей: незашифрованной части, в которой содержится исходный

текст T, и зашифрованной части, представляющей собой цифровую подпись. Цифровая подпись S вычисляется с помощью закрытого ключа (D, n) по формуле:

$$S = T^D \bmod n.$$

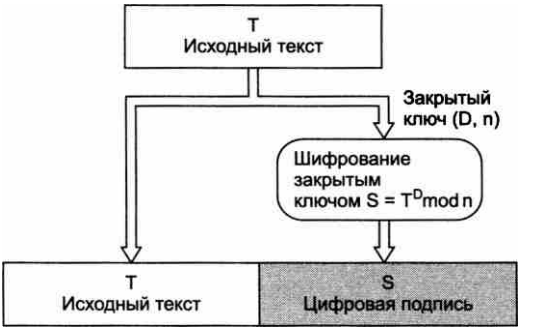


Рис. 27.7. Схема формирования цифровой подписи по алгоритму RSA

Сообщение посылается в виде пары (T, S). Каждый пользователь, имеющий соответствующий открытый ключ (E, n), получив сообщение, отделяет открытую часть T, расшифровывает цифровую подпись S и проверяет равенство:

$$T = S^E \bmod n.$$

Если результат расшифровки цифровой подписи совпадает с открытой частью сообщения, считается, что документ подлинный, не претерпел никаких изменений в процессе передачи, а автором его является именно тот человек, который передал свой открытый ключ получателю.

К недостаткам данного алгоритма можно отнести то, что длина подписи в этом случае равна длине сообщения, что не всегда удобно. Для уменьшения «длины» электронной подписи вместо $S = T^D \bmod n$ используется формула:

$$S = (H(T))^D \bmod n.$$

Здесь $H(T)$ — хеш-функция, преобразующая исходное сообщение в короткий дайджест. В этом случае получатель сообщения (T, S) должен сначала применить к открытому тексту T хеш-функцию H и получить дайджест $H(T)$, а затем приступить к расшифровке подписи S открытым ключом. Если расшифрованная подпись совпадает с дайджестом, то авторство сообщения доказано. Использование хеш-функций дает выигрыш не только в объеме сообщения, но и во времени получения электронной подписи.

Если помимо проверки аутентичности документа, обеспечиваемой цифровой подписью, надо обеспечить его конфиденциальность, то после применения к тексту цифровой подписи перед передачей его по каналу связи выполняют совместное шифрование исходного текста и цифровой подписи любым совместно выбранным способом шифрования.

Аутентификация на основе цифровых сертификатов

Аутентификация с применением цифровых сертификатов является альтернативой применению паролей и представляется естественным решением в условиях, когда число пользователей сети (пусть и потенциальных) измеряется миллионами. В таких обстоятельствах процедура предварительной регистрации пользователей, связанная с назначением и хранением их паролей, становится крайне обременительной, опасной, а иногда и просто нереализуемой. При наличии сертификатов сеть, которая дает пользователю доступ к своим ресурсам, не хранит никакой информации о своих пользователях — они ее предоставляют сами в своих запросах в виде сертификатов, удостоверяющих личность пользователей. Сертификаты выдаются специальными уполномоченными сертифицирующими организациями (CO) — **центрами сертификации** (Certificate Authority, CA), или **удостоверяющими центрами**. Поэтому задача хранения секретной информации (закрытых ключей) возлагается на самих пользователей, что делает это решение гораздо более масштабируемым, чем вариант с централизованной базой паролей.

Сертификат представляет собой электронную форму, в которой содержится следующая информация:

- открытый ключ владельца данного сертификата;
- сведения о владельце сертификата, такие, например, как имя, адрес электронной почты, наименование организации, в которой он работает, и т. п.;
- наименование сертифицирующей организации, выдавшей данный сертификат;
- электронная подпись сертифицирующей организации, то есть зашифрованные закрытым ключом этой организации данные, содержащиеся в сертификате.

Использование сертификатов основано на предположении, что сертифицирующих организаций не много и их открытые ключи широко доступны, например, из публикаций в журналах.

Сертификаты могут быть представлены в трех формах (рис. 27.8):

- *в открытой форме* сертификат содержит всю информацию в незашифрованном виде;
- *в форме из двух частей* — открытой, содержащей всю информацию в незашифрованном виде, и закрытой, представляющей собой открытую часть, зашифрованную закрытым ключом сертифицирующей организации;
- *в форме из трех частей* — во-первых, открытой, во-вторых, открытой, но зашифрованной закрытым ключом сертифицирующей организации, в-третьих, части, представляющей собой первые две части, зашифрованные закрытым ключом владельца.

Когда пользователь хочет подтвердить свою личность, он предъявляет свой сертификат в двух формах: открытой (то есть такой, в какой он получил его в сертифицирующей организации) и зашифрованной с применением своего закрытого ключа. Сторона, проводящая аутентификацию, берет из незашифрованного сертификата открытый ключ пользователя и расшифровывает с его помощью зашифрованный сертификат. Совпадение результата с открытым сертификатом подтверждает, что предъявитель действительно является владельцем закрытого ключа, соответствующего указанному открытому.

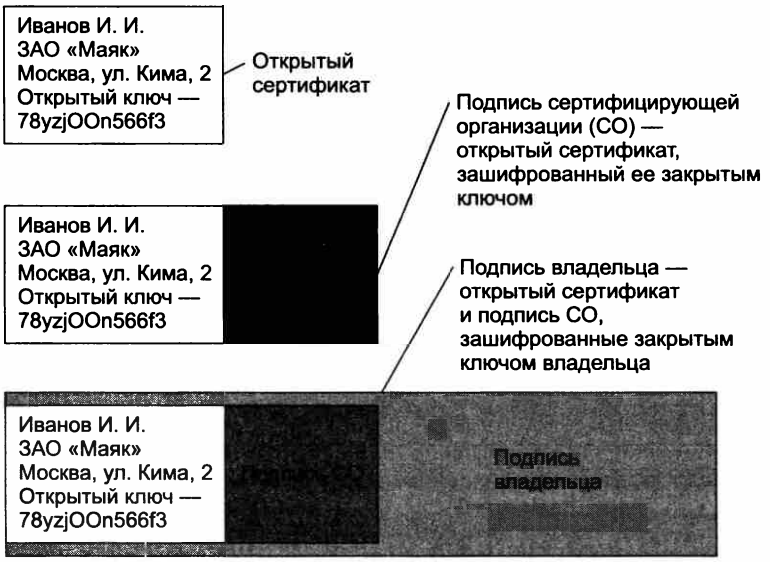


Рис. 27.8. Формы представления цифрового сертификата

Затем с помощью известного открытого ключа указанной в сертификате организации проводится расшифровка подписи этой организации в сертификате. Если в результате получается тот же сертификат с тем же именем пользователя и его открытым ключом, значит, он действительно прошел регистрацию в сертификационном центре, является тем, за кого себя выдает, и указанный в сертификате открытый ключ действительно принадлежит ему.

Сертификаты можно применять не только для аутентификации, но и *для предоставления прав доступа к ресурсам*. Для этого в сертификат могут вводиться дополнительные поля, в которых указывается принадлежность его владельцев к той или иной категории пользователей. Эта категория назначается сертифицирующей организацией в зависимости от условий, на которых выдается сертификат. Например, организация, поставляющая через Интернет на коммерческой основе информацию, может выдавать сертификаты определенной категории пользователям, оплатившим годовую подписку на некоторый бюллетень, тогда веб-сервер будет предоставлять доступ к страницам бюллетеня только пользователям, предъявившим сертификат данной категории.

Подчеркнем тесную связь открытых ключей с сертификатами. Сертификат является удостоверением не только личности, но и принадлежности открытого ключа.

Цифровой сертификат устанавливает и гарантирует соответствие между открытым ключом и его владельцем.

Это предотвращает угрозу подмены открытого ключа. Если некоторый абонент *А* получает по сети сертификат от абонента *Б*, то он может быть уверен, что открытый ключ, содержащийся в сертификате, гарантированно принадлежит абоненту *Б*, адрес и другие сведения о котором содержатся в этом сертификате. Это значит, что абонент *А* может без опасений использовать открытый ключ абонента *Б* для секретных посланий в адрес последнего.

При наличии сертификатов отпадает необходимость хранить на серверах корпораций списки пользователей с их паролями, вместо этого достаточно иметь на сервере список имен и открытых ключей сертифицирующих организаций. Может также понадобиться некоторый механизм для установления соответствия категорий владельцев сертификатов традиционным группам пользователей, чтобы можно было в неизменном виде задействовать механизмы управления избирательным доступом большинства операционных систем или приложений.

Сертификат является средством аутентификации пользователя при его обращении к сетевым ресурсам, роль аутентифицирующей стороны играют при этом информационные серверы корпоративной сети или Интернета. В то же время и сама процедура получения сертификата также включает этап аутентификации, когда аутентификатором выступает сертифицирующая организация. Для получения сертификата клиент должен сообщить сертифицирующей организации свой открытый ключ и те или иные сведения, удостоверяющие его личность. Все эти данные клиент может отправить по электронной почте или принести на съемном носителе лично. Перечень необходимых данных зависит от типа получаемого сертификата. Сертифицирующая организация проверяет доказательства подлинности, помещает свою цифровую подпись в файл, содержащий открытый ключ, и посылает сертификат обратно, подтверждая факт принадлежности данного конкретного ключа конкретному лицу. После этого сертификат может быть встроен в любой запрос на использование информационных ресурсов сети (рис. 27.9).

Практически важным является вопрос о том, кто имеет право выполнять функции сертифицирующей организации. Во-первых, задачу обеспечения своих сотрудников сертификатами может взять на себя само предприятие. В этом случае упрощается процедура первичной аутентификации при выдаче сертификата. Предприятия достаточно осведомлены о своих сотрудниках, чтобы брать на себя задачу подтверждения их личности. Для автоматизации процесса генерации, выдачи и обслуживания сертификатов предприятия могут использовать готовые программные продукты: например, компания Netscape Communications выпустила сервер сертификатов, который организации могут у себя устанавливать для выпуска своих сертификатов. Во-вторых, эти функции могут выполнять независимые центры по выдаче сертификатов, работающие на коммерческой основе, например сертифицирующий центр компании Verisign. Сертификаты компании Verisign выполнены в соответствии с международным стандартом X.509 и используются во многих продуктах, ориентированных на защиту данных, в том числе в популярном протоколе защищенного канала SSL. Любой желающий может обратиться с запросом на получение сертификата на веб-сервер этой компании.

Механизм получения пользователем сертификата хорошо автоматизируется в сети в модели клиент-сервер, когда браузер исполняет роль клиента, а в сертифицирующей организации установлен специальный сервер выдачи сертификатов. Браузер генерирует для пользователя пару ключей, оставляет закрытый ключ у себя и передает частично заполненную форму сертификата серверу. Для того чтобы не подписанный еще сертификат нельзя было подменить при передаче по сети, браузер ставит свою электронную подпись, зашифровывая сертификат выработанным закрытым ключом. Сервер сертификатов подписывает полученный сертификат, фиксирует его в своей базе данных и возвращает его каким-либо способом владельцу. Очевидно, что при этом может выполняться еще и неформальная процедура подтверждения пользователем своей личности и права на получение сертификата, требующая участия оператора сервера сертификатов. Это могут быть доказательства

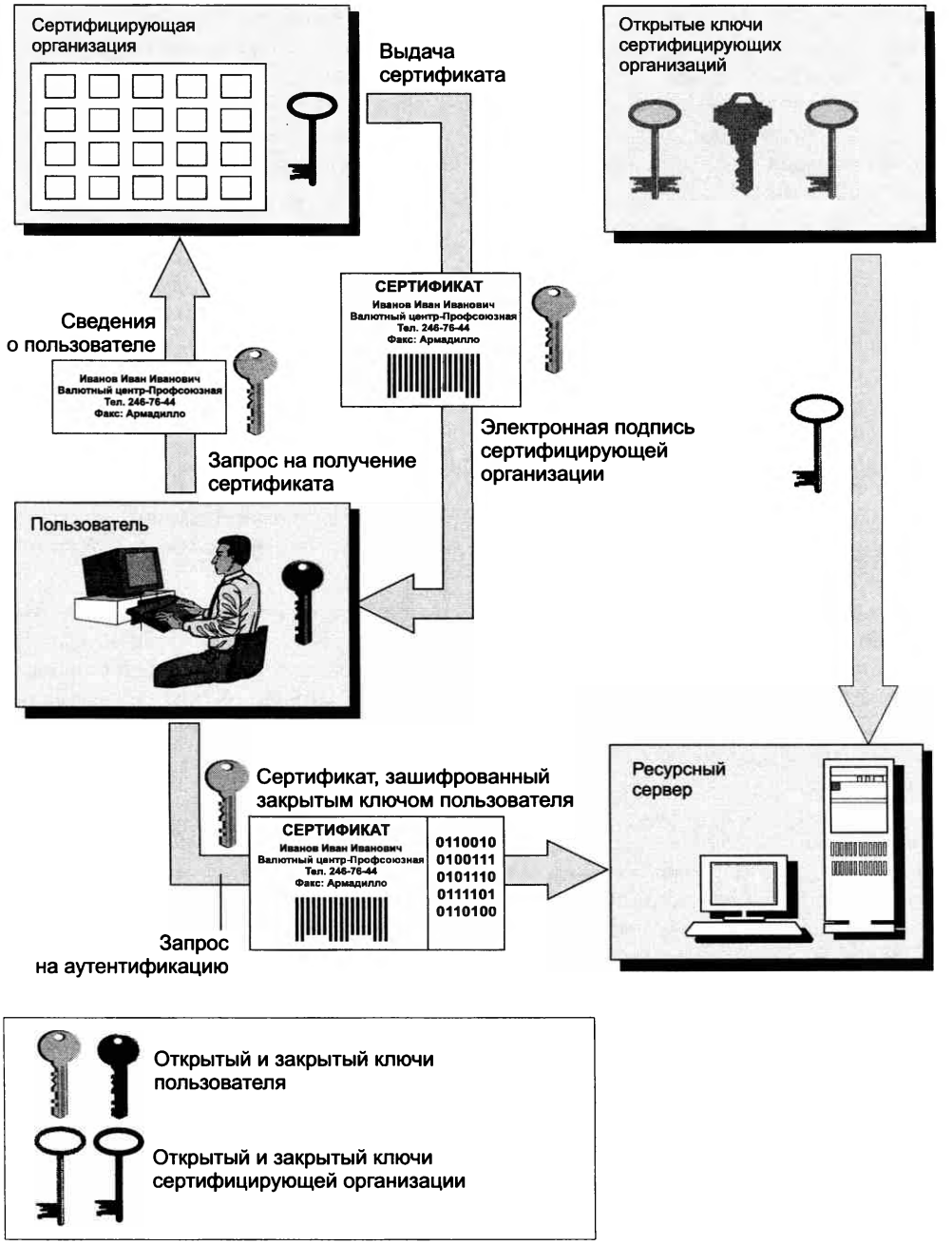


Рис. 27.9. Схема аутентификация пользователей на основе сертификатов

оплаты услуги, доказательства принадлежности к той или иной организации — все случаи жизни предусмотреть и автоматизировать нельзя. После получения сертификата браузер сохраняет его вместе с закрытым ключом и использует при аутентификации на тех серверах, которые поддерживают такой процесс. В настоящее время существует большое количество протоколов и продуктов, применяющих сертификаты. В частности, практически все браузеры и операционные системы реализуют поддержку сертификатов.

Несмотря на активное использование технологии цифровых сертификатов во многих системах безопасности, эта технология еще не решила целый ряд серьезных проблем. Это, прежде всего, поддержание базы данных о выпущенных сертификатах. Сертификат выдается не навсегда, а на некоторый вполне определенный срок. По истечении срока годности сертификат должен либо обновляться, либо аннулироваться. Кроме того, необходимо предусмотреть возможность досрочного прекращения полномочий сертификата. Все заинтересованные участники информационного процесса должны быть вовремя оповещены о том, что некоторый сертификат уже недействителен. Для этого сертифицирующая организация должна оперативно поддерживать список отозванных сертификатов.

Имеется также ряд проблем, связанных с тем, что сертифицирующие организации существуют не в единственном числе. Все они выпускают сертификаты, но даже если эти сертификаты соответствуют единому стандарту (сейчас это, как правило, стандарт X.509), все равно остаются нерешенными многие вопросы. Все ли сертифицирующие центры заслуживают доверия? Каким образом можно проверить полномочия того или иного сертифицирующего центра? Можно ли создать иерархию сертифицирующих центров, когда сертифицирующий центр, стоящий выше, мог бы сертифицировать центры, расположенные в иерархии ниже? Как организовать совместное использование сертификатов, выпущенных разными сертифицирующими организациями?

Для решения этих и многих других проблем, возникающих в системах, использующих технологии шифрования с открытыми ключами, оказывается необходимым комплекс программных средств и методик, называемый *инфраструктурой с открытыми ключами* (Public Key Infrastructure, PKI).

Рассмотренная схема аутентификации включает три основных элемента: это пользователи, цифровые сертификаты и центры сертификации. Для того чтобы данная схема работала надежно и эффективно, в нее должны быть включены дополнительные элементы, которые в совокупности с основными и образуют PKI.

В число дополнительных элементов может входить, например, регистрационный центр (Registration Authority), который служит посредником между пользователем, запросившим сертификат, и центром сертификации. Пользователь обычно обращается к регистрационному центру с помощью веб-интерфейса и сообщает данные о себе. Регистрационный центр проверяет эту информацию и в случае ее подлинности передает данные о пользователе, подписанные собственным закрытым ключом, центру сертификации. Регистрационный центр может обслуживать несколько центров сертификации. При отсутствии регистрационного центра его функции выполняет центр сертификации.

Другим типом дополнительных элементов PKI являются разнообразные хранилища сертификатов, содержащие информацию о действующих, отозванных и истекших сертификатах.

Аутентификация программных кодов

Электронная подпись и сертификаты могут применяться для доказательства аутентичности (подлинности) программ. Пользователю важно быть уверенным, что программа, которую он загрузил с какого-либо сервера Интернета, действительно содержит коды, разработанные определенной компанией. Компания Microsoft предложила для этих целей технологию **аутентикода** (authenticode).

Организация, желающая подтвердить свое авторство на программу, должна встроить в распространяемый код так называемый **подписывающий блок** — аутентикод (рис. 27.10). Этот блок состоит из двух частей. Первая часть — это сертификат организации — разработчика данной программы, полученный обычным образом от какого-либо сертифицирующего центра. Вторую часть образует зашифрованный дайджест, полученный в результате применения хеш-функции к распространяемому коду. Шифрование дайджеста выполняется с помощью закрытого ключа организации.

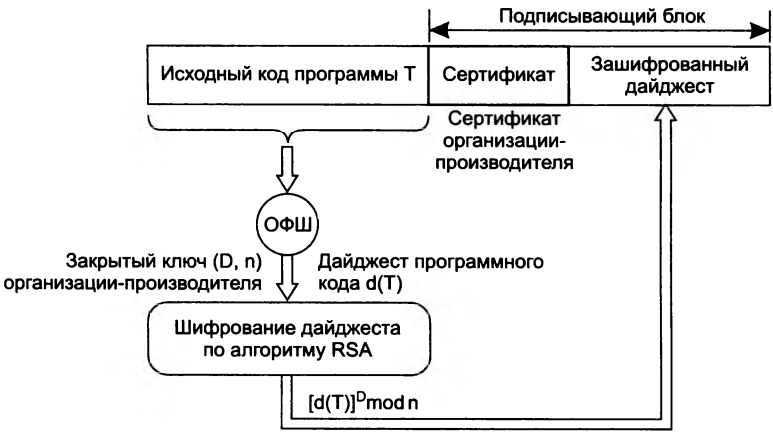


Рис. 27.10. Схема получения аутентикода

Компания-разработчик может потребовать от пользователя программы доказательство легальности ее приобретения. Для этого компания может запросить регистрационный номер программы (Product ID или Serial Number), называемый также *лицензионным ключом активации*. Обычно этот номер пишется на отдельном бланке, прилагаемом к поставляемой программе, наносится на упаковку или высылается по электронной почте при покупке программы через Интернет.

Другим способом доказательства легальности приобретения и законности использования программных продуктов являются миниатюрные электронные устройства — *электронные замки*, подобные уже рассмотренным нами аппаратным аутентификаторам. Эти устройства поставляются вместе с защищаемыми от нелегального использования программами. Перед запуском программы электронный замок должен быть подключен к компьютеру, например через USB-порт. Инициирующий блок программы обращается к данному устройству с запросом и, получив «правильный» ответ, начинает работать, если же ожидаемый ответ не поступает, то выполнение программы блокируется. Таким образом, электронный замок

действует как специфический аутентификатор пользователя, доказывающий то, что он является законным владельцем программы.

(S) *Биометрическая аутентификация*

Технологии управления доступом и авторизации

После того как пользователь, пройдя аутентификацию, доказал свою легальность, ему предоставляется некоторый набор прав по отношению к защищаемым системой ресурсам.

> Наделение легальных пользователей правами доступа к ресурсам называется **авторизацией**. Процедура приведения авторизации в действие называется **управлением доступом** (access control).

Если, например, субъект пытается использовать ресурс с запрещенным для него типом доступа, то механизм управления доступом должен отклонить эту попытку и, возможно, уведомить систему об этом инциденте с целью генерации сигнала тревоги.

Формы представления ограничений доступа

При решении задачи управления доступом необходимо руководствоваться принципом минимальных привилегий. В соответствии с ним каждому субъекту в системе должен быть назначен минимально возможный набор прав, достаточный для решения ровно тех задач, на которые он уполномочен. Применение этого принципа ограничивает те возможные потери, которые могут быть понесены в результате неумышленных ошибок или неавторизованных действий.

Решение о наделении пользователей правами (или, что одно и то же, об ограничении их прав на доступ к ресурсам) основывается на политике безопасности предприятия и может формулироваться в разных формах.

Ограничение доступа может задаваться в форме **правил**. На основании правила система управления доступом в любой момент времени *динамически* решает вопрос о предоставлении или непредоставлении доступа. Правило может строиться с учетом различных факторов, в том числе длительности сеанса связи (ограничение доступа по времени использования ресурса), возраста человека (ограничение для детей на доступ к некоторым сайтам), времени суток (разрешение на использование ресурсов и сервисов Интернета только в рабочие часы). Популярной мерой ограничения доступа в Интернет является *капча* (capcher), в этом случае субъекту, обратившемуся с запросом к ресурсу, предлагается ввести символы, выведенные на экран в таком искаженном виде, в котором их сможет распознать только человек — таким образом исключается доступ к ресурсам искусственных субъектов (программных систем). Другим распространенным правилом является правило, которое носит специальное название — *необходимо знать* (need-to-know). В соответствии с этим правилом каждый сотрудник имеет право доступа только к тем информационным ресурсам, которые ему необходимы для выполнения его служебных обязанностей.

Для ограничения доступа используются также **контентно-** и **контекстно-зависимые правила**. Например, в компании может быть принято правило, что некоторым категориям пользователей запрещается доступ к документам, содержащим те или иные ключевые слова или фразы, такие как «для ограниченного использования», «секретно» или кодовое название проекта. Ограничения могут быть наложены на доступ к ресурсам, содержащим текст на иностранном языке. Это были примеры контентно-зависимых правил. В контекстно-зависимых правилах принимаются во внимание некоторые факторы, характеризующие текущее состояние среды и/или предысторию (контекст) запроса. Простейшим правилом такого рода является отказ в доступе пользователю, который сделал подряд три безуспешных попытки аутентификации. Или доступ к некоторому сетевому ресурсу предприятия может быть запрещен, если к моменту текущего обращения пользователь выполнил несколько обращений к внешнему сайту, содержимое которого не связано напрямую с его профессиональными интересами.

Эффективным средством ограничения доступа является **конфигурирование пользовательского интерфейса**. Таким путем пользователь может быть лишен не только возможности обращаться к тем или иным каталогам и файлам, но и возможности видеть на своем экране часть структуры файловой системы, доступ к которой ему запрещен. Администратор может настроить систему меню пользовательского интерфейса так, что некоторые пункты этих меню не будут выводиться на экран, что исключит принципиальную возможность запуска пользователем части функций.

Матрица прав доступа является универсальной и наиболее гранулированной (то есть тонко дифференцированной) формой представления политики контроля доступа, она прямо «в лоб» описывает для каждого пользователя набор конкретных операций, которые ему разрешается выполнять по отношению к каждому объекту (рис. 27.11).

Субъекты / Объекты	User 1	User 2	User 3
File 1	Читать и записывать	Читать и записывать	Читать
File 2	Читать и записывать	Нет доступа	Читать
File 3	Записывать	Нет доступа	Читать

Рис. 27.11. Матрица прав доступа

Матричный способ описания прав доступа теоретически дает возможность отразить все многообразие отношений субъектов и объектов системы для всех возможных сочетаний {субъект, объект, назначенные права}. Однако этот универсальный способ представления, как правило, очень сложно реализовать на практике из-за громоздкости матрицы, учитывая огромное число элементов — как субъектов, так и объектов — в вычислительной системе.

Особенностью матрицы прав доступа является не только ее большая размерность, но и наличие большого числа *нулевых* элементов. Такой вид матриц в математике называют разряженными. Нулевое значение здесь говорит о том, что для данного сочетания {субъект, объект} права доступа не определены, а именно такие сочетания составляют большинство

в реальных системах. Свойство разреженности матрицы может быть использовано для более компактного представления правил доступа.

С каждым объектом можно связать **список управления доступом** (Access Control List, ACL), в котором указаны только те субъекты (пользователи), которые имеют разрешение на доступ к данному объекту (ресурсу). Ясно, что количество субъектов в данном списке будет значительно меньше общего числа субъектов системы. Такие списки должны быть созданы для всех ресурсов. Способ описания прав доступа набором списков столь же универсальный и гибкий, как матрица, но вместе с тем имеет более компактный вид, так как он не включает пустые элементы матрицы. Список ACL состоит из **элементов управления доступом** (Access Control Element, ACE), каждый из которых описывает права доступа определенного пользователя к данному ресурсу. На рис. 27.12 список ACL состоит из трех элементов ACE.

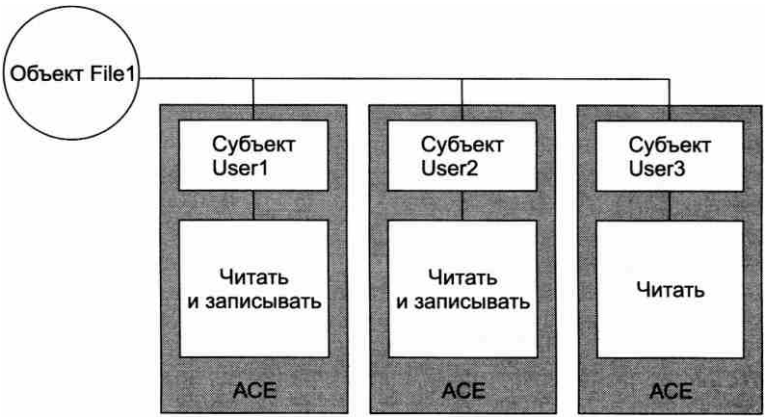


Рис. 27.12. Список управления доступом к объекту

Права доступа могут быть определены как по отношению к ресурсам, так и по отношению к пользователям. В последнем случае его называют **списком разрешений** (capability). На рис. 27.13 показан список разрешений, которые имеет пользователь User 2 по отношению к ресурсам File 1, File 2 и File 3.

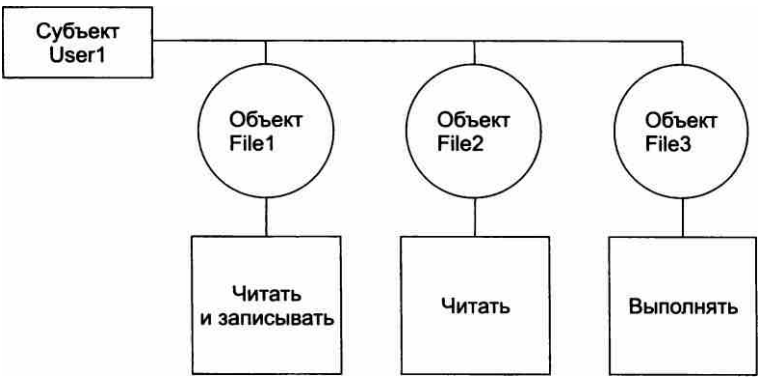


Рис. 27.13. Список разрешений пользователя User 2

Очевидно, что совокупность списков управления доступом ко всем ресурсам системы несет ту же самую информацию, что и совокупность списков разрешений для всех пользователей, так как и те и другие являются разными проекциями одной и той же матрицы. В одних реализациях систем управления доступом (например, в большинстве операционных систем) применяются ограничения, заданные для объекта (ACL), а в других (например, в некоторых расширениях системы Kerberos, включающих авторизацию) — ограничения для субъекта (списки разрешений).

Другим способом «сжатия» матрицы является определение прав доступа для групп субъектов по отношению к группам объектов. Такое представление возможно, когда многие элементы матрицы имеют одинаковое значение, что соответствует ситуации в реальной системе, когда некоторая группа пользователей имеет одинаковые права. Это дает возможность компактно описать права доступа с помощью матрицы меньшей размерности.

В некоторых случаях, если существует простое правило определения прав доступа, хранение матрицы вообще не требуется, значения элементов матрицы могут вычисляться системой управления доступом *динамически*. Например, пусть все объекты и субъекты системы изначально снабжены метками из одного и того же множества. Кроме того, предположим для простоты изложения, что для всех объектов определен только один вид операции доступа. И пусть существует правило: доступ к объекту разрешен, если метки субъекта и объекта совпадают, и не разрешен, если не совпадают. Имея такое правило, нет смысла заранее создавать и хранить матрицу — проще вычислять соответствующий элемент при каждой попытке доступа.

Ранее мы рассматривали различные подходы к хранению и представлению информации о правах доступа, не придавая значения тому, каким образом они были назначены. Однако способ назначения прав — авторизация — существенно влияет на способ управления доступом.

Существует два основных подхода к авторизации:

- для авторизации выделяется особый *полномочный орган* (authority), который принимает все решения о наделении пользователей правами относительно всех объектов;
- функции принятия решений по авторизации *делегируются* некоторым субъектам.

Как видим, управление доступом может быть реализовано множеством различных способов, отражающих разные подходы к заданию и приведению в исполнение ограничений, однако большинство реализуемых на практике способов может быть отнесено к одной из следующих категорий:

- **дискреционный метод доступа** (Discretionary Access Control[1], DAC), называемый также избирательным, или произвольным;
- **мандатный метод доступа** (Mandatory Access Control[2], MAC), называемый также принудительным;
- **ролевой доступ** (Role-based Access Control, RBAC), называемый также недискреционным методом доступа (nondiscretionary access control).

Помимо этих методов, взятых «в чистом виде», система управления доступом может базироваться на их комбинации.

---

[1] Discretionary — действующий по своему усмотрению.

[2] Mandatory — обязательный, принудительный.

Дискреционный метод управления доступом

Одно из первых систематических изложений принципов DAC было предпринято в 1987 году в документе NCSC-TG-003-87, «Руководство по дискретному управлению доступом». В то время модель DAC была самой распространенной схемой управления доступом, таковой она остается и по сегодняшний день — большинство универсальных ОС реализуют дискреционную модель. В документе NCSC дается следующее определение метода DAC:

> Дискреционный метод представляет собой средство ограничения доступа к объектам, базирующееся на уникальных идентификаторах субъекта и/или групп, к которым этот субъект относится. Управление доступом в методе DAC является дискреционным, или произвольным, в том смысле, что субъект, обладающий некоторыми разрешениями на доступ к объектам, может по своему усмотрению передать часть своих полномочий (иногда прямо, а иногда — опосредованно) другим субъектам.

Отсюда следуют две главные особенности дискреционного метода:

- Права доступа в методе DAC описываются в виде списков ACL, которые дают возможность гибкого и гранулированного определения набора разрешенных операций для каждого отдельного пользователя по отношению к каждому отдельному ресурсу, причем и пользователи и ресурсы задаются уникальными идентификаторами.
- В методе DAC право назначать права на доступ к объектам делегируются отдельным пользователям — владельцам объектов. То есть им разрешается действовать «по своему усмотрению» и назначать другим пользователям права на доступ к тем объектам, владельцами которых они являются.

Таким образом, процедура авторизации является распределенной между множеством пользователей-владельцев. Владельцами считаются пользователи, создавшие объект, или пользователи, которые были назначены владельцами другими уполномоченными на то пользователями или системными процессами. Владелец имеет полный контроль над созданным им объектом и несет всю полноту ответственности за управление доступом к нему. Вместе с тем он может назначать права доступа к своим объектам, руководствуясь некоторым правилом, принятым на предприятии.

Основным достоинством метода DAC является его гибкость, обусловленная свободой пользователей наделять правами или аннулировать права других пользователей на доступ к своим ресурсам, а также возможностями тонкой настройки набора разрешенных операций. Однако это достоинство имеет свою оборотную сторону. Как и всякая распределенная система, система управления доступом по методу DAC страдает от *невозможности гарантированно проводить общую политику*, осуществлять надежный контроль действий пользователей. Любая политика безопасности, принятая на предприятии, может быть нарушена в результате ошибочных или вредительских действий пользователей.

Другой недостаток дискреционного метода связан с тем, что здесь *права на доступ определяются по отношению к объекту*, а не к его содержимому. Это означает, что любой пользователь (точнее, его процесс), имеющий доступ к файлу согласно некоторому списку ACL1, может скопировать его содержимое в другой файл, характеризуемый другим списком ACL2. Это показывает, что системы с контролем доступа по методу DAC не могут применяться там, где требуется очень высокий уровень защиты информации.

(S) *Слабость метода DAC*

Мандатный метод управления доступом

Мандатный доступ позволяет реализовать системы, отвечающие самым строгим требованиям безопасности, как правило, они используются в правительственных и военных учреждениях или в других организациях, для которых чрезвычайно важен высокий уровень защиты данных.

К основным чертам мандатного метода управления доступом можно отнести следующие:

- авторизацию и управление доступом осуществляет центральный полномочный орган, отвечающий за безопасность (обычно в роли такого органа выступает операционная система);
- решение о предоставлении права доступа принимается операционной системой динамически на основе простого правила, которое разрабатывается уполномоченными на то лицами на основе политики безопасности.

Простота правил достигается тем, что как субъекты, так и объекты разбиваются на небольшое число групп. Каждой группе объектов присваивается **уровень (гриф) секретности**, а группам субъектов — **уровни допуска** к объектам того или иного уровня секретности. В разных системах могут быть приняты разные правила, но все они базируются на сравнении уровня секретности объекта и уровня допуска субъекта. Например, правило может быть следующим: субъекту разрешается доступ к объекту, если уровень его допуска равен уровню или выше уровня секретности объекта. На рис. 27.14 это правило представлено в виде матрицы.

Уровень допуска субъектов / Уровень секретности объектов	Уровень от совершенно секретно и ниже	Уровень от секретно и ниже	Уровень данных для служебного пользования
Совершенно секретно	Доступ разрешен	0	0
Секретно	Доступ разрешен	Доступ разрешен	0
Данные для служебного пользования	Доступ разрешен	Доступ разрешен	Доступ разрешен

Рис. 27.14. Правило мандатного доступа, представленное в виде матрицы

Пользователи должны принимать решение системы как данность, они лишены возможности управлять доступом к своим ресурсам или передавать свои права другим пользователям. В отличие от систем DAC, мандатный доступ *имеет централизованный характер и позволяет жестко проводить принятую политику безопасности.*

Элементы, описывающие уровни секретности объектов или уровни допуска субъектов, называют **метками безопасности** (security labels). Мандатный метод управления доступом предусматривает назначение меток безопасности всем без исключения субъектам и объектам системы, чтобы в дальнейшем они использовались системой для принятия решения о допуске.

В большинстве случаев для адекватного отражения политики безопасности невозможно сформулировать правило, основанное на учете только уровней секретности и допусков. К одному и тому же уровню секретности могут быть отнесены самые разные материалы, а в соответствии с принципом минимальных привилегий пользователь должен получать доступ только к той информации, которую ему необходимо знать.

Для того чтобы сделать возможным более специфическое задание прав доступа, в метки безопасности объекта и субъекта добавляется информация о конкретном виде данных, к которому относится данный объект или к которому разрешен доступ данному субъекту соответственно.

Таким образом, каждая метка безопасности состоит из двух частей (рис. 27.15):

- часть, отражающая уровень секретности/ допуска, называется *классификацией*;
- часть, характеризующая специфику информации, называется *категорией*.

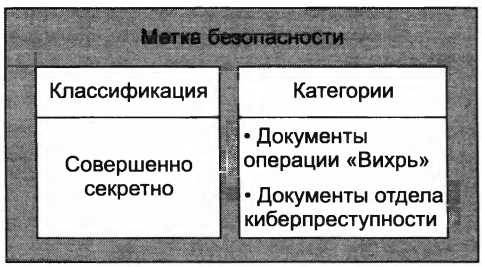


Рис. 27.15. Структура метки безопасности объекта/субъекта

Категория относит данные к определенному виду информации. Например, разные категории могут быть присвоены материалам, относящимся к разным проектам, разным административным подразделениям, разным профессиональным группам. Одному и тому же объекту/субъекту может быть присвоено несколько категорий. Так, отчет о завершении этапа некоторой антитеррористической операции может быть отнесен не только к категории материалов, касающихся данной операции, но и дополнительно к категории материалов подразделения, занимающегося этой работой. Объекты одной категории могут быть классифицированы по-разному: например, одна часть отнесена к более высокому уровню секретности, а другая часть — к более низкому.

Уровни секретности/допуска, которых обычно не много, образуют иерархию от наивысшего до самого низкого уровня. Субъект, имеющий допуск к некоторому уровню, получает его и по отношению ко всем нижележащим уровням.

Правило, определяющее право доступа, строится на анализе обеих частей меток безопасности объекта и субъекта. Доступ разрешается, если выполняются следующие два условия:

- классификация субъекта равна или выше классификации объекта;
- по меньшей мере одна из категорий объекта, к которому пытается получить доступ субъект, совпадает хотя бы с одной из категорий данного субъекта.

Рисунок 27.16 иллюстрирует соотношение между классификацией и категорией. Здесь разная закраска кружков служит для обозначения разных категорий объектов. На рисунке показано три уровня классификации: «совершенно секретно», «секретно» и «для служебного

пользования». Объекты одной категории могут принадлежать разным уровням классификации. В метке безопасности субъекта указана классификация «секретно» и перечислены две категории, к которым ему разрешен доступ. Стрелками показаны три попытки доступа. Попытка обращения к уровню «совершенно секретно» была заблокирована системой из-за недостаточно высокого уровня допуска субъекта. Обращение к объекту уровня «секретно» было разрешено, так как классификация субъекта равна классификации объекта, а категория объекта совпала с одной из категорий, указанных в метке безопасности субъекта. Попытка доступа к объекту уровня «для служебного пользования» была пресечена, хотя субъект и имеет более высокий уровень допуска («секретно»). В данном случае ограничением служит категория объекта, которая не совпадает ни с одной из категорий субъекта.

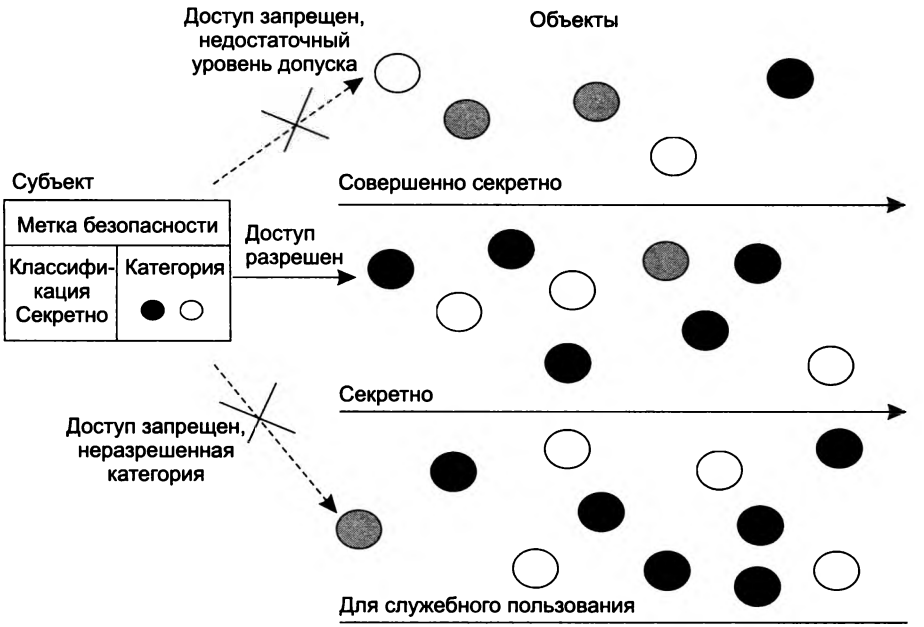


Рис. 27.16. Правило мандатного доступа

Мандатный доступ, как уже отмечалось, является более безопасным, чем дискреционный, но для его эффективной реализации требуется большой объем подготовительной работы, а после запуска системы необходимо поддерживать в актуальном состоянии метки безопасности существующих объектов, а также назначать метки новым ресурсам и пользователям.

Ролевое управление доступом

Метод управления доступом RBAC, основанный на ролях, по сравнению с методами DAC и MAC более приближен к реальной жизни. Как видно из названия, основным его свойством является использование «ролей». Понятие «роль» в данном контексте ближе всего к понятию «должность» или «круг должностных обязанностей». Поскольку одну и ту же должность могут занимать несколько людей, то и одна и та же роль может быть приписана разным пользователям.

Роли устанавливаются для целей авторизации. Набор ролей в системе RBAC должен некоторым образом (не однозначно) соответствовать перечню различных должностей, существующих на предприятии, к которому эта система относится. Система RBAC лучше всего работает в организациях, в которых существует четкое распределение должностных обязанностей.

Разрешения приписываются ролям, а не отдельным пользователям или группам пользователей (рис. 27.17). А уже затем те или иные роли приписываются пользователю. Например, в системе управления доступом, развернутой в банке, всем юристам приписана роль «юрист», трейдерам — роль «трейдер», менеджерам — роль «менеджер» и т. д. Процесс определения ролей должен включать тщательный анализ того, как функционирует организация, какой набор функций должен выполнять работник, имеющий ту или иную должность. Каждой из ролей назначаются права доступа, необходимые и достаточные пользователям для выполнения служебных обязанностей, обусловленных приписыванием к данной роли.

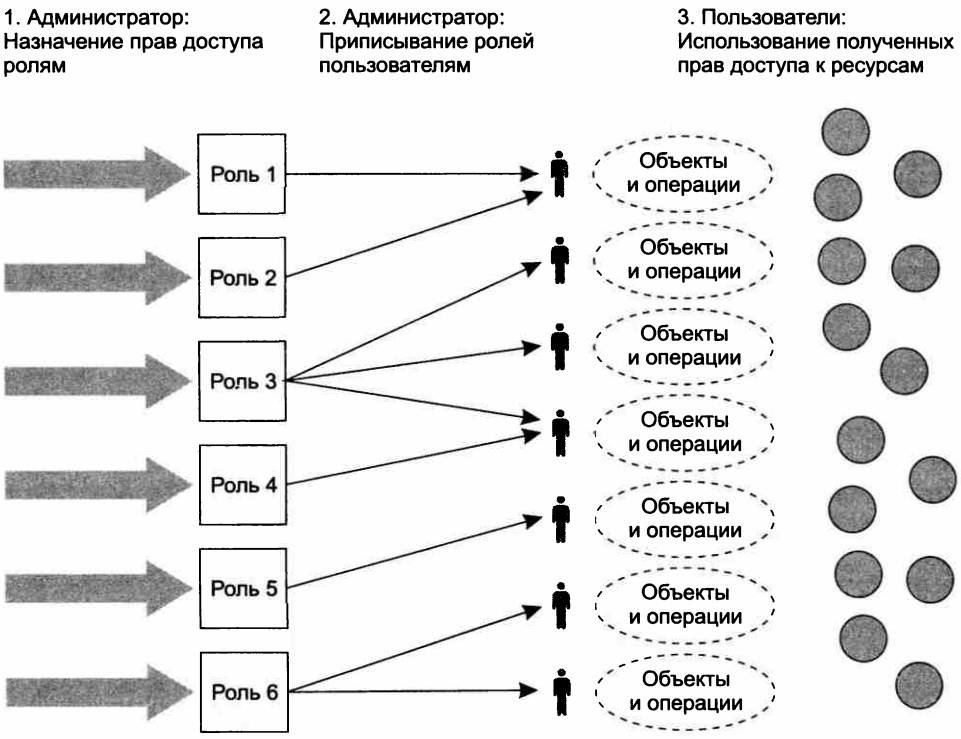


Рис. 27.17. Схема авторизации в системах управления доступом на основе ролей

Каждому пользователю может быть приписано несколько ролей (с некоторыми ограничениями, о которых рассказано далее). Во время сеанса работы пользователя все роли, которые ему назначены, становятся активными и он получает права доступа, являющиеся результатом объединения прав доступа всех этих групп.

Все пользователи, играющие одну и ту же роль, имеют идентичные права. Изменение производственной ситуации — расширение бизнеса, внедрение новых технологий, продвижение сотрудника по служебной лестнице или перевод в другое подразделение и др. — все это может вызвать аннулирование одной роли пользователя и приписывание ему другой роли. Такой подход упрощает администрирование прав доступа: вместо необходимого в методах DAC и MAC отслеживания и обновления прав каждого отдельного пользователя в методе RBAC достаточно изменить роль или заменить одну роль другой.

Таким образом, в системе RBAC имеются удобные механизмы для соблюдения принципа минимальных привилегий. И хотя теоретически метод DAC позволяет проводить еще более тонкую настройку прав пользователя, практически невозможно проконтролировать этот процесс так, чтобы добиться реализации этого принципа. В системе, где механизм назначения прав распределен между всеми пользователями, очень сложно отследить ситуацию, когда набор прав пользователя становится не адекватным решаемым им задачам.

Согласно природе производственных отношений, должностные обязанности сотрудников, занимающих разные позиции, могут частично перекрываться. Некоторые самые общие функции, такие, например, как ознакомление с инструкциями по соблюдению режима работы предприятия, резервирование отпусков, фиксирование на внутреннем сайте компании индивидуального рабочего графика и др., могут быть обязательными для всех сотрудников. Применительно к ролям это означает, что администратор должен выполнять много рутинной работы по приписыванию одних и тех же прав доступа разным ролям, в том числе вновь создаваемым. Решением этой проблемы является иерархическая организация ролей, когда одна роль может включать другую роль, тем самым расширяя свой набор прав за счет добавления прав, ассоциированных с инкапсулированной ролью.

Иерархия ролей создается определением для них отношений, называемых наследованием: в соответствии с этим определением если роль R2 является наследницей R1, то все права роли R1 приписываются к правам роли R2, а все пользователи роли R2 приписываются к пользователям роли R1 (рис. 27.18). Таким образом, установление отношений наследования является еще одним способом наделения пользователя правами наряду с явным назначением пользователю некоторой роли.

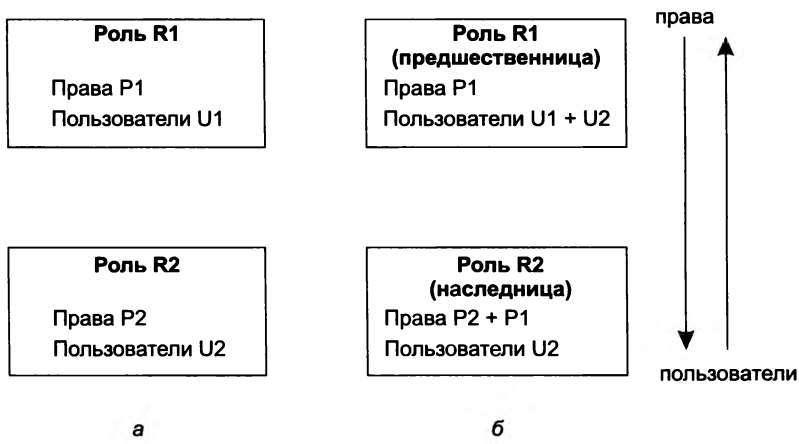


Рис. 27.18. Отношение наследования ролей: *а* — независимые роли; *б* — роль R2 является наследницей роли R1

Отношения наследования относятся к типу «многие ко многим», то есть у одной роли может быть несколько наследниц и одна роль может быть наследницей нескольких ролей.

Иерархия ролей обычно в той или иной степени отражает структуру реального предприятия. На рис. 27.19 показан фрагмент организационной структуры предприятия.

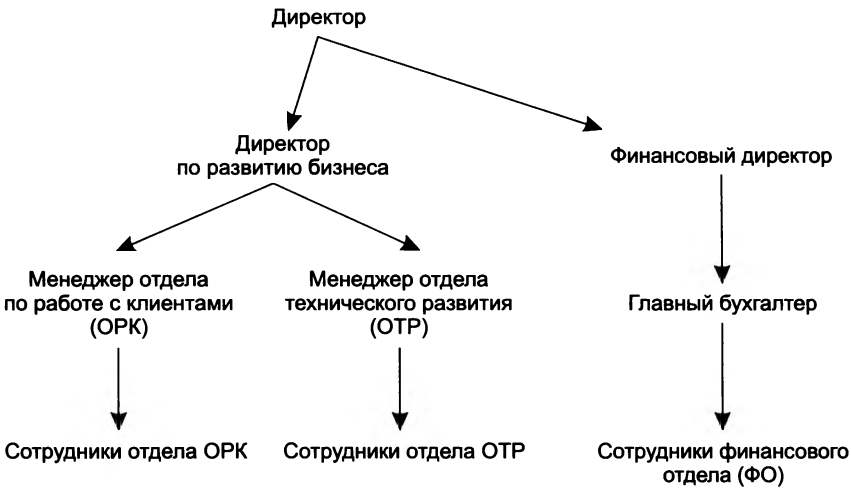


Рис. 27.19. Фрагмент организационной структуры предприятия

Такой *организационной структуре* может быть поставлена в соответствие *ролевая структура*, полученная в результате установления между ролями отношений наследования (рис. 27.20). Роль «сотрудник» представляет собой общие, наличествующие у всех сотрудников организации права. Должность менеджера по работе с клиентами добавляет к должностным обязанностям рядового сотрудника отдела ОРК еще ряд функций. Например, менеджер обязан разрабатывать план увеличения клиентской базы, что требует доступа к некоторым финансовым документам. Находящийся с ним на одном уровне менеджер отдела развития (ОР) также нуждается в расширении прав доступа к информационным ресурсам по отношению к рядовым сотрудникам отдела ОР. После установления отношений наследования с ролями «сотрудник ОРК», «сотрудник ОТР», «сотрудник ФО» и «директор» все общие права сотрудников оказались неявным образом добавлены к этим ролям-наследницам, а их пользователи соответственно переместились наверх. Наследники следующей ступени — роли «менеджер ОРК» и «менеджер ОТР» — сами являются предшественниками для роли «директор по развитию», которая, таким образом, аккумулировала права этих двух ролей.

Важным положением безопасности является *принцип разделения обязанностей*, в соответствии с которым определенные должностные функции не должны поручаться одному и тому же человеку. К примеру, сотрудник, которому назначена роль «инженер», побывав в командировке, должен после возвращения составить финансовый отчет о своих тратах. Затем этот отчет должен быть проверен и представлен к оплате, эти действия возлагаются на сотрудника, отнесенного к роли «сотрудник финансового отдела». Понятно, что такое совмещение функций, то есть одновременная принадлежность одного пользователя к ролям «инженер» и «сотрудник финансового отдела», является нежелательным. Чтобы

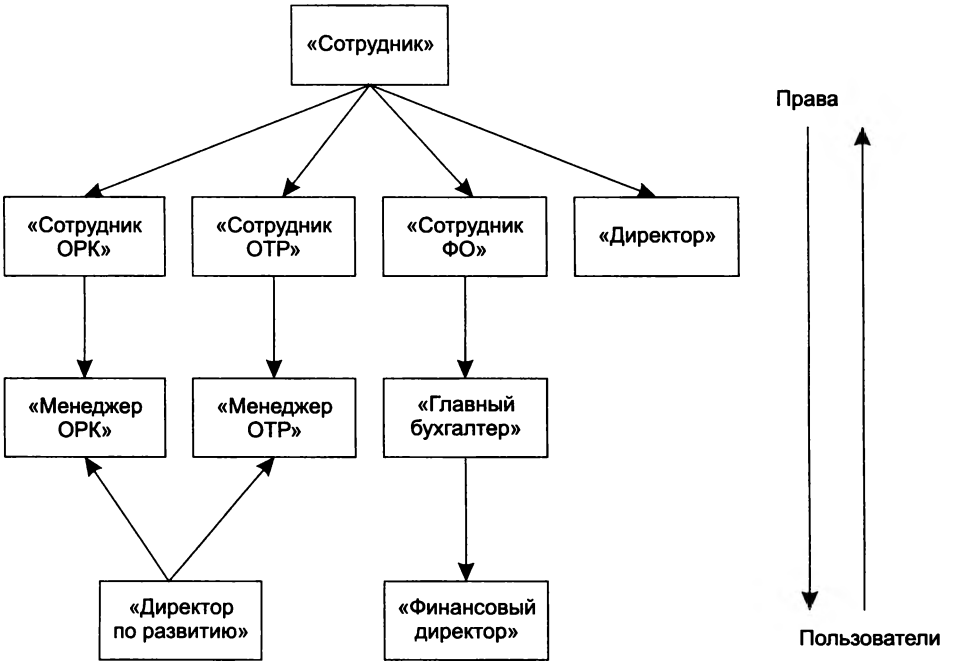


Рис. 27.20. Структура ролей, образованная отношениями наследования

избежать подобных ситуаций, в методе RBAC предусмотрен специальный механизм, накладывающий ограничения на приписывание ролей пользователям. Этот механизм действует следующим образом. Совокупность ролей, относительно совмещения которых нужно устанавливать ограничения, объединяется в устойчивую группу, и к ней приписывается число-ограничитель. В нашем случае это группа ролей {«инженер», «сотрудник финансового отдела»}, которой должен быть приписан ограничитель 1. Если теперь администратором будет сделана попытка приписать пользователю обе эти роли, то система заблокирует его действия.

Итак, к характерным особенностям ролевого управления доступом можно отнести следующее:

- RBAC сочетает в себе черты мандатного и дискреционного способов управления доступом.
- Ролевую систему управления доступом легче администрировать и контролировать, чем дискреционную. В DAC права назначаются пользователю «мелкими порциями», что позволяет ему выполнять ту или иную конкретную операцию над отдельным объектом (запись в определенный файл, чтение другого файла, запуск некоторой программы). Такой способ помогает с ювелирной точностью создавать и индивидуально настраивать комплекс прав доступа пользователя, однако он является очень трудоемким, вследствие чего возрастает возможность ошибок. В RBAC права доступа выдаются в виде «глыбы» — интегрированного набора разрешений, рассчитанных на возможность выполнения некоторых относительно сложных операций: заполнение кредитного документа, генерация отчетов и др.

- RBAC является централизованным методом управления доступом — так же, как и в мандатном методе, пользователь лишен возможности управлять назначением прав. Назначение пользователю роли можно считать некоторым аналогом приписывания уровня допуска пользователю мандатной системы. Однако RBAC является более гибким способом, чем MAC, по возможностям настройки прав доступа он находится ближе к DAC.

Метод RBAC нельзя отнести к хорошо масштабируемым. Он эффективно работает в пределах единой системы или приложения, таких, например, как FreeBSD, Solaris, СУБД Oracle, MS Active Directory, но на больших предприятиях, имеющих тысячи сотрудников, поддержание множества ролей с их отношениями наследования становится сложной и запутанной задачей. Занимая промежуточное положение между мандатным и дискреционным методами, ролевое управление доступом уступает им обоим в масштабируемости. В мандатном методе централизованный характер принятия решений (который не способствует масштабируемости) компенсируется простотой выполняемого алгоритма назначения прав. В дискреционном же методе, напротив, сложность механизма наделения правами компенсируется распределенным характером процедуры принятия решений.

Системы аутентификации и управления доступом операционных систем

Аутентификации пользователей ОС

Существует две принципиально отличные схемы аутентификации, реализуемые операционными системами и специальными сетевыми службами. В одной из них, которую мы будем называть *локальной системой аутентификации*, операционная система работает в пределах одного компьютера: она задействует базу аутентификационных данных пользователей компьютера с этой ОС, и результаты аутентификации могут применяться только для доступа к ресурсам этого компьютера.

По другой схеме работает так называемая *система аутентификации домена*: она базируется на центральной базе аутентификационных данных пользователей группы компьютеров (*домена аутентификации*), хранящейся на одном из серверов сети, и результаты аутентификации служат для доступа к ресурсам данного домена.

Локальная система аутентификации ОС работает при *логическом входе пользователя* как с терминала *компьютера*, так и *через* сеть. Первый вариант называют **интерактивным** логическим входом, второй — **удаленным** (сетевым, или неинтерактивным). Понятно, что при удаленном логическом входе риски безопасности выше, так как аутентификационные данные передаются через сеть — корпоративную или Интернет — и их легче перехватить. Перехват данных аутентификации представляет собой угрозу даже в случае строгой аутентификации, когда пароль не передается в открытом виде по сети или же не передается вовсе: при наличии большого массива аутентификационных данных, то есть данных перехватов большого количества процедур входа одного и того же пользователя, пароль может быть вычислен по имеющимся результатам его ввода.

Хотя локальные системы аутентификации ОС поддерживают все распространенные методы аутентификации — на основе многоразовых и одноразовых паролей (аппаратных

и программных), биометрических данных и цифровых сертификатов, — основным методом аутентификации пользователей является метод на базе многоразового пароля. Практически все универсальные ОС, такие как MS Windows, Unix/Linux и MAC OS X, предлагают этот метод по умолчанию.

Одноразовые пароли, обеспечивающие более надежную аутентификацию, чем многоразовые, чаще применяются при удаленном логическом входе через соединения VPN с шифрованием информации, где передача аутентификационной информации идет через Интернет и, следовательно, риск ее перехвата и взлома особенно велик. Одноразовые пароли могут сочетаться с многоразовыми при двухфакторной аутентификации.

Аутентификация на основе сертификатов применяется чаще всего для удаленно работающих пользователей, которые предъявляют сертификаты, выданные сервером сертификации организации, к которой принадлежит пользователь.

Аутентификация на основе биометрических данных штатными средствами универсальных ОС обычно не поддерживается, так как их повышенная надежность нужна только в особо защищенных системах и, кроме того, для поддержки биометрической аутентификации требуется приобрести и установить соответствующее специальное программное обеспечение и специальные устройства.

Необходимо отличать процедуру аутентификации пользователя операционной системы от процедуры аутентификации пользователя серверной части некоторого приложения. Многие серверные приложения имеют собственную систему аутентификации пользователей, никак не связанную с системой аутентификации ОС, под управлением которой они работают. Например, так работают многие реализации FTP-сервера, сервера баз данных. Независимость системы аутентификации сервера приложений имеет как свои положительные, так и отрицательные стороны. Преимуществом здесь является разграничение по умолчанию пользователей ОС, которым потенциально может понадобиться доступ к любому ресурсу компьютера, и пользователей некоторого сервиса, которым нужен доступ только к ресурсам, относящимся к данному сервису, например только к файлам, хранящимся в корневом каталоге FTP-сервера. К недостаткам же можно отнести низкую защищенность протокола аутентификации некоторых приложений, а также необходимость запоминания двух различных имен и паролей для одного и того же пользователя, ошибки администраторов ОС и сервисов из-за дублирования учетных записей и т. п.

Аутентификация в ОС семейства Unix. Протокол SSH

Рассмотрим особенности процедуры аутентификации на примере ОС семейства Unix. Традиционный интерактивный вход с алфавитно-цифрового терминала (или программы, его эмулирующей) с аутентификацией по паролю по-прежнему является основным способом логического входа в систему ОС Unix. Утилита **login** поддерживает процедуру *интерактивного входа* пользователей в текстовом режиме[1]. Эта процедура включает

[1] С появлением графических интерфейсов пользователя, таких как GNOME KDE, текстовый режим работы утилиты login был дополнен ее графическим вариантом, однако суть ее работы от этого не изменилась.

локальную аутентификацию пользователей на основе их учетных данных, хранящихся в особом файле на диске.

Удаленный вход пользователей Unix во времена сравнительно безопасного Интернета выполнялся с помощью протокола **telnet**, который является протоколом эмуляции текстового терминала поверх транспортных средств стека TCP/IP. Протокол telnet в открытом виде передает символы, набираемые пользователем, и ответы ОС, поэтому перехватить пароль, набираемый клиентом telnet, не составляет труда. Из-за незащищенности telnet его строго не рекомендуется задействовать для аутентификации пользователей Unix через Интернет; да и крупная корпоративная сеть также может представлять опасность для такой передачи.

Основным современным средством ОС Unix для *удаленного входа* является многофункциональный пакет программ **SSH** (*Secure Shell*), который поддерживает различные методы защищенной аутентификации, а также некоторые виды защищенных операций с файлами через сеть. За годы его существования было разработано много как открытых, так и коммерческих версий. Сегодня наиболее распространенной является версия *Open SSH* v.2, клиент и сервер которой включены практически во все свободно распространяемые версии Unix/Linux: Fedora, CentOS, Ubuntu и др.

Протокол SSH работает в архитектуре клиент-сервер, серверная часть представлена демоном sshd, клиентская часть — утилитой ssh, которая выполняет запрос пользователя на логический вход в удаленный хост. Например, для логического входа в хост ganimede.co.uk пользователь victor выполняет ssh-команду `victor@ganymede.co.uk`.

Протокол SSH обеспечивает взаимную аутентификацию клиента и сервера, а также безопасную передачу данных между ними за счет шифрования. Шифрование выполняется на основе техники открытых и закрытых ключей. Аутентификация с помощью SSH может осуществляться различными способами, которые могут быть сведены к одному из следующих двух подходов.

- Открытый и закрытый ключи клиента генерируются автоматически и в дальнейшем прозрачным для пользователя способом применяются для шифрования передаваемых данных. Аутентификация выполняется на основе многоразового пароля или по алгоритму строгой аутентификации. Пароль и дайджест от пароля, а также другие аутентификационные данные передаются по безопасному каналу.
- Открытый и закрытый ключи используются не только для шифрования данных, но и для аутентификации клиента. Применяя утилиту ssh-keygen, пользователь вручную генерирует открытый и закрытый ключи, помещает каждый из них в отдельный файл на своем клиентском хосте, а затем копирует файл с открытым ключом в свой домашний каталог на хосте-сервере, в который он хочет удаленно входить. После этого пользователь может выполнять удаленный вход на этот сервер по протоколу SSH без пароля. В целях аутентификации клиента сервер с помощью известного ему открытого ключа данного клиента предпринимает попытку расшифровать информацию, поступающую к нему в ходе процедуры установления соединения. Поскольку эту информацию клиент шифрует своим закрытым ключом, то успех расшифровки означает успех аутентификации.

Семейство ОС Unix поддерживает не только локальную схему аутентификации на основе файла паролей, хранящегося на диске данного компьютера, но и централизованные системы, такие как NIS, NIS+, Open Directory, Kerberos.

Управление доступом в операционных системах

Что касается систем управления доступом в универсальных ОС, то там доминирует дискреционная модель управления доступом; согласно этой модели, владелец ресурса (пользователь, который его создал или которому передано владение) самостоятельно определяет, кто имеет доступ к этому ресурсу и какие операции с ним он может выполнять. Практически все популярные сегодня семейства универсальных ОС — Unix/Linux/CentOS/Ubuntu, Mac OS X, MS Windows — опираются на дискреционную модель доступа как на основную.

Мандатная модель — это особенность специализированных ОС, рассчитанных на применение в среде с повышенными требованиями к безопасности. Тем не менее существует набор модулей ядра Linux под названием SELinux (Security Enhanced Linux), который реализует многие свойства мандатной модели в среде Linux.

Модель ролевого управления доступом применяется в универсальных ОС частично в виде механизма встроенных групп с предопределенными правами, сосуществуя с моделью дискреционного доступа для индивидуальных пользователей и групп.

В качестве примера рассмотрим некоторые свойства системы управления доступом в ОС семейства Windows. Эта система, построенная на основе *дискреционного алгоритма*, отличается высокой степенью гибкости, которая достигается за счет большого разнообразия типов субъектов и объектов доступа, а также детализации операций доступа.

Для разделяемых ресурсов в ОС семейства Windows применяется общая модель объекта, который содержит такие характеристики безопасности, как набор допустимых операций, идентификатор владельца, список управления доступом. Объекты создаются для любых ресурсов в том случае, когда они являются или становятся разделяемыми: файлов, каталогов, устройств, секций памяти, процессов.

Для системы безопасности ОС семейства Windows характерно наличие большого количества различных предопределенных (встроенных) субъектов доступа — как отдельных пользователей, так и групп. Так, в системе всегда имеются пользователи Administrator, System и Guest, а также группы Users, Administrators, Account Operators, Server Operators, Everyone и др. Смысл этих встроенных пользователей и групп состоит в том, что они изначально наделены некоторыми правами, облегчая администратору работу по созданию эффективной системы разграничения доступа. При добавлении нового пользователя администратору остается только решить, к какой группе или группам отнести этого пользователя. Конечно, администратор может создавать новые группы, а также добавлять права к встроенным группам для реализации собственной политики безопасности, но во многих случаях встроенных групп оказывается вполне достаточно.

ОС семейства Windows поддерживает три класса операций доступа, которые отличаются типом субъектов и объектов, участвующих в этих операциях.

- **Разрешения** (permissions) — это множество операций, которые могут быть определены для субъектов всех типов по отношению к объектам любого типа: файлам, каталогам, принтерам, секциям памяти и т. д. Разрешения по своему назначению соответствуют правам доступа к файлам и каталогам в ОС Unix.
- **Права** (user rights) определяются для субъектов типа группа на выполнение некоторых системных операций: установку системного времени, архивирование файлов, выключение компьютера и т. п. В этих операциях участвует особый объект доступа — операционная система в целом.

- **Возможности пользователей** (user abilities) определяются для отдельных пользователей на выполнение действий, связанных с формированием их операционной среды: например, изменение состава главного меню программ, возможность пользоваться пунктом меню Run (Выполнить) и т. п. За счет уменьшения набора возможностей (доступных пользователю по умолчанию) администратор может «заставить» пользователя работать с той операционной средой, которая наилучшим образом соответствует политике безопасности.

В основном именно права, а не разрешения отличают одну встроенную группу пользователей от другой. Права у встроенных групп могут быть встроенными либо изменяемыми. **Встроенные права** являются неотъемлемыми атрибутами встроенных групп, администратор не может ими распоряжаться. **Изменяемые права** можно удалять или добавлять к правам встроенной группы из общего списка изменяемых прав. Например, встроенная группа Users не имеет никаких изменяемых или встроенных прав, в то время как группа Administrators наделена широким набором как встроенных (создание и управление пользовательской учетной информацией, управление аудитом, форматирование жесткого диска сервера и др.), так и изменяемых прав (интерактивный и удаленный логический вход, установление прав собственности на файлы, управление аудитом событий, связанных с безопасностью, изменение системного времени, останов системы и др.).

Для ОС семейства Windows характерна высокая степень детализации операций доступа. Так, для доступа к файлам и каталогам предусмотрено два типа разрешений:

- **индивидуальные разрешения** относятся к элементарным операциям;
- **стандартные разрешения** являются объединением индивидуальных разрешений.

В табл. 27.2 приведен перечень индивидуальных разрешений на доступ к файлам.

Таблица 27.2. Индивидуальные разрешения для каталогов и файлов

Разрешение	Для файла
Read (R)	Чтение данных, атрибутов, имени владельца и разрешений файла
Write (W)	Чтение имени владельца и разрешений файла, изменение атрибутов файла, изменение и добавление данных файла
Execute (X)	Чтение атрибутов файла, имени владельца и разрешений. Выполнение файла, если он хранит код программы
Delete (D)	Удаление файла
Change Permission (P)	Изменение разрешений файла
Take Ownership (O)	Вступление во владение файлом

Для файлов определено четыре стандартных разрешения: No Access, Read, Change и Full Control, соответствие которых группам индивидуальных разрешений показано в табл. 27.3.

Таблица 27.3. Стандартные и индивидуальные разрешения для файлов

Стандартное разрешение	Индивидуальные разрешения
No Access	Ни одного
Read	RX
Change	RWXD
Full Control	Все

Результатом успешной аутентификации пользователя в ОС Windows является создание для него системой так называемого **токена доступа** (access token). Токен доступа привязывается ко всем процессам, которые данный пользователь создает в течение сеанса работы. Токен доступа включает идентификатор пользователя и идентификаторы всех групп, в которые он входит, список прав пользователя на выполнение системных операций и др.

Доступ к объекту описывается списком ACL. Владелец объекта — обычно пользователь, который его создал, — обладает правом управлять доступом к объекту и может изменять ACL объекта, чтобы позволить или не позволить другим осуществлять тот или иной вид доступа к объекту.

Проверка прав доступа к объектам любого типа выполняется *централизованно* с помощью модуля ОС — **монитора безопасности** (Security Reference Monitor). Монитор безопасности сравнивает идентификаторы пользователя и групп пользователей из токена доступа процесса с соответствующими идентификаторами, хранящимися в элементах ACL объекта. Система безопасности могла бы осуществлять проверку разрешений каждый раз, когда процесс использует объект. Но список ACL состоит из многих элементов, процесс в течение своего существования может иметь доступ ко многим объектам, и количество активных процессов в каждый момент времени также велико. Поэтому для снижения издержек проверка выполняется только при первом обращении процесса к объекту[1], а не при каждом использовании объекта.

Гибкость системы безопасности ОС семейства Windows во многом определяется разнообразием прав на выполнение системных действий, высокой степенью детализации операций доступа к объектам, а также существованием встроенных групп, позволяющих администратору эффективно реализовывать политику безопасности данной информационной системы.

Централизованные системы аутентификации и авторизации

Концепция единого логического входа

Традиционный способ аутентификации с помощью многоразовых паролей отлично подходит для случая, когда пользователь все время работает с единственным компьютером, обращаясь только к его ресурсам и ресурсам Интернета, не требующим аутентификации. Такому пользователю нужно запоминать и периодически менять только один пароль. К сожалению, такая ситуация редко встречается в жизни. Более типичным является случай, когда пользователю приходится работать на разных, географически рассредоточенных компьютерах — рабочем стационарном компьютере, домашнем стационарном компьютере, личном планшете, гостевом компьютере предприятия-партнера — и при этом получать доступ к различным серверам, например северам своего предприятия, серверам предприятия-партнера, к защищенным веб-сайтам Интернета.

В том случае, когда каждый компьютер и каждый сервер требуют отдельной аутентификации с помощью многоразового пароля, пользователю приходится помнить и обновлять

[1] Точнее, права доступа проверяются при каждом открытии объекта.

довольно много паролей, и с этой задачей многие пользователи справляются не очень успешно. Согласно исследованию, проведенному Network Applications Consortium, около 70 % звонков пользователей в службу ИТ-поддержки связано с просьбой восстановления забытого пароля, а в среднем пользователь крупной корпоративной сети тратит на процедуры логического входа 44 часа в год.

Неудивительно, что большие усилия затрачиваются на разработку процедур **единого логического входа** (Single Sign On, SSO).

> Целью единого логического входа является создание такого порядка аутентификации, при котором пользователь выполняет вход в сеть только один раз, доказывая свою аутентичность с помощью любого способа аутентификации, а затем результат этой аутентификации прозрачным для пользователя способом применяется каждый раз, когда ему нужно доказывать свою аутентичность какому-либо серверу или приложению.

В настоящее время не существует системы аутентификации, реализующей концепцию единого логического входа, которая бы работала со всеми типами операционных систем, приложений и при этом учитывала бы разнообразные отношения между организациями, к которым принадлежат пользователи и информационные ресурсы. Однако имеются системы, позволяющие организовать единый логический вход для однородной в каком-то отношении информационной системы, например для сети, использующей только одну определенную ОС или один определенный протокол аутентификации, либо для группы организаций, доверяющих друг другу при аутентификации своих пользователей. Так, например, свойство однородности операционных систем является условием применимости системы единого входа на основе справочной службы Microsoft Active Directory.

Обобщенная схема, иллюстрирующая идею систем единого логического входа, представлена на рис. 27.21.

Рис. 27.21. Схема единого логического входа

В этой схеме имеются три элемента:

- *Пользователь* располагает некоторой информацией, достаточной для его аутентификации. Это может быть информация любого типа из упомянутых ранее — многоразовый пароль, одноразовый пароль, цифровой сертификат, биометрические данные и т. п. На рисунке в качестве примера показан вариант аутентификации на основе многоразового пароля, здесь ID — идентификатор, PW — *многоразовый пароль*.
- *Провайдер* идентичности (Identity Provider) — это система, которая может аутентифицировать пользователя на основе базы данных учетных записей пользователей. Этот элемент может иметь и другие названия, например *сервер аутентификации*.
- *Сервис-провайдер* (Service Provider), называемый также *ресурсным сервером*, — это система, предоставляющая сервисы пользователям. Такими сервисами могут быть файловый сервис, почтовый сервис, веб-сервис, сервис баз данных и т. п. Предполагается, что сервис предоставляется только аутентифицированным пользователям.

Особенностью этой схемы является то, что провайдер сервисов не поддерживает базу учетных данных пользователей. База учетных данных имеется только у провайдера идентичности, а провайдер сервисов доверяет результатам аутентификации пользователей, выполненной провайдером идентичности. Говорят, что в таком случае существуют *доверительные отношения* (trust relationships) между провайдером идентичности и провайдером сервисов.

Пользователь выполняет логический вход в сеть, обращаясь к провайдеру идентичности. Если пользователь смог подтвердить свою аутентичность, то провайдер идентичности предоставляет пользователю некоторую информационную структуру — *токен доступа*, который пользователь хранит в своей базе данных. При необходимости получения доступа к некоторому сервису пользователь предъявляет токен доступа ресурсному серверу. Токен доступа защищен криптографически таким образом, что ресурсный сервер имеет возможность проверить тот факт, что токен был выдан пользователю сервером аутентификации, которому ресурсный сервер доверяет аутентифицировать пользователей. Говорят, что в этом случае происходит *вторичная аутентификация* пользователя, но для самого пользователя она прозрачна, так как предъявлением токена доступа занимается программное обеспечение его компьютера.

Токен доступа обычно имеет ограниченное время действия, например сутки, поэтому пользователь должен его возобновлять, повторяя процедуру с сервером аутентификации.

(S) *Классификация систем единого логического входа*

Система Kerberos

Kerberos — это сетевая служба, предназначенная для централизованного решения задач аутентификации в крупных сетях. Kerberos реализует процедуру единого логического входа в пределах домена, где клиенты и серверы поддерживают этот протокол.

Система централизованной аутентификации тесно связана с системой централизованного управления доступом, так как последняя должна основываться на результатах аутентификации каждый раз, когда вычислительный процесс, представляющий пользователя, пытается получить доступ к ресурсу компьютера, входящего в некоторый домен.

Система Kerberos может работать в среде многих популярных ОС, например в ОС семейства Windows система Kerberos встроена как основной компонент безопасности. Существуют реализации Kerberos для семейства Unix, включая Red Hat Linux, Fedora, Centos, Ubuntu, и для Mac OS X. Первая версия Kerberos была разработана для проекта Athena в Массачусетском технологическом институте. Текущей версией является версия 5, которая стандартизована IETF в RFC 4120.

В основе функционирования этой достаточно громоздкой системы лежит несколько простых принципов:

- в сетях, использующих систему безопасности Kerberos, все процедуры аутентификации между клиентами и серверами сети выполняются через посредника, которому доверяют обе стороны процесса аутентификации, причем таким авторитетным арбитром является сама система Kerberos;
- в системе Kerberos клиент должен доказывать свою аутентичность для доступа к каждой службе, услуги которой он запрашивает;
- все обмены данными в сети выполняются в защищенном виде с применением симметричного алгоритма шифрования AES (или DES в ранних реализациях Kerberos).

Сетевая служба Kerberos построена в архитектуре клиент-сервер, что позволяет ей работать в самых сложных сетях. Kerberos-клиент устанавливается на всех компьютерах сети, которые могут обратиться к какой-либо сетевой службе. В таких случаях Kerberos-клиент от лица пользователя передает запрос на Kerberos-сервер и поддерживает с ним диалог, необходимый для выполнения функций системы Kerberos.

Итак, в системе Kerberos имеются следующие участники: Kerberos-сервер, Kerberos-клиенты, ресурсные серверы (рис. 27.22). Kerberos-клиенты пытаются получить доступ к сетевым ресурсам — файлам, приложениям, принтеру и т. д., находящимся на ресурсных серверах. Этот доступ может быть предоставлен, во-первых, только легальным пользователям, а во-вторых, при наличии у них достаточных полномочий, определяемых службами авторизации соответствующих ресурсных серверов — файловым сервером, сервером приложений, сервером печати. Однако в системе Kerberos ресурсным серверам запрещается «напрямую» принимать запросы от клиентов, им разрешается начинать рассмотрение запроса клиента только тогда, когда на это поступает разрешение от Kerberos-сервера. Таким образом, путь клиента к ресурсу в системе Kerberos состоит из трех этапов:

1. Определение легальности клиента, логический вход в сеть, получение разрешения на продолжение процесса получения доступа к ресурсу.
2. Получение разрешения на обращение к ресурсному серверу.
3. Получение разрешения на доступ к ресурсу.

Для решения первой и второй задач клиент обращается к Kerberos-серверу. Каждая из этих двух задач решается отдельным сервером, входящим в состав Kerberos-сервера. Выполнение первичной аутентификации и выдача разрешения на продолжение процесса получения доступа к ресурсу осуществляются так называемым *Kersberos-сервером аутентификации* (Authentication Server, AS). Этот сервер хранит в своей базе данных информацию об идентификаторах и паролях пользователей. Пароли пользователей, а точнее хеш-функции от паролей, являются секретными ключами пользователей.

Вторую задачу, связанную с получением разрешения на обращение к ресурсному серверу, решает другая часть Kerberos-сервера — *Kerberos-сервер квитанций* (Ticket-Granting Server,

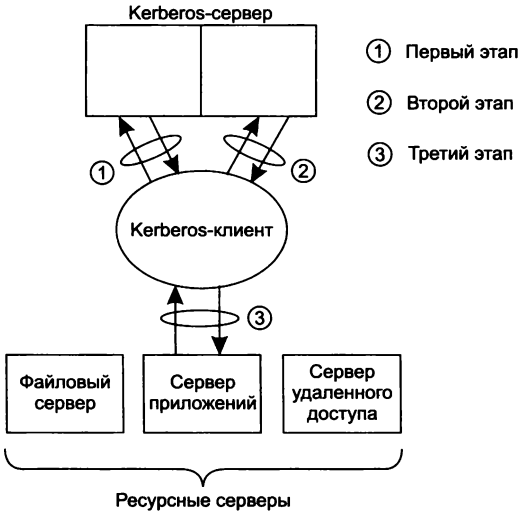


Рис. 27.22. Три этапа работы системы Kerberos

TGS). Сервер квитанций для легальных клиентов выполняет дополнительную проверку и дает клиенту разрешение на доступ к нужному ему ресурсному серверу, для чего наделяет его электронной формой-квитанцией. Для выполнения своих функций сервер квитанций использует копии секретных ключей всех ресурсных серверов, которые хранятся у него в базе данных. Помимо этих ключей TGS-сервер имеет еще один секретный ключ, общий с AS-сервером.

Третья задача — получение разрешения на доступ непосредственно к ресурсу — решается на уровне ресурсного сервера собственными средствами, *не относящимися* непосредственно к системе Kerberos, но способными взаимодействовать с ней.

Секретные ключи пользователей и ресурсных серверов образуют базу данных ключей Kerberos-сервера. Собственно, обладание секретным ключом и является условием аутентификации пользователя или ресурсного сервера. Помимо секретных ключей пользователей и ресурсных серверов в Kerberos также применяются секретные ключи сеансов аутентификации, которые распределяет Kerberos-сервер. Из-за этого обстоятельства Kerberos-сервер также называется *Kerberos Key Distribution Centre*, или *Kerberos KDC*. Секретные ключи пользователей и ресурсных серверов называют еще мастер-ключами, так как они являются постоянными ключами, аутентифицирующими субъект, в отличие от ключей сеансов, которые имеют непродолжительный срок действия.

Введение центра аутентификации существенно улучшает масштабируемость системы аутентификации на основе симметричного шифрования по сравнению с децентрализованной системой. Действительно, если на предприятии имеется N пользователей и M ресурсных серверов, которым нужна взаимная аутентификация (мы предполагаем, что пользователям взаимная аутентификация не нужна), то при децентрализованной аутентификации необходимо $N \times M$ ключей, что для предприятия с 1000 сотрудников и 50 ресурсными серверами дает 50 000 ключей. При централизованной системе аутентификации необходимо иметь только $N + M$ ключей, что равно 1050 ключей для нашего примера, — то есть почти в 50 раз меньше.

Необходимо подчеркнуть, что Kerberos обеспечивает защищенную аутентификацию сторон только в начальный момент сеанса обмена данными между ними. После этого защита данных — их конфиденциальность, аутентичность и целостность — должна обеспечиваться средствами ресурсного сервера и клиента, если это необходимо.

ПРИМЕЧАНИЕ

При описании протоколов взаимодействия Kerberos-клиента и Kerberos-сервера, а также Kerberos-клиента и ресурсного сервера использован термин «квитанция» (ticket), означающий в данном случае электронную форму, выдаваемую Kerberos-сервером клиенту, которая играет роль некоего удостоверения личности и разрешения на доступ к ресурсу.

Первичная аутентификация

Процесс доступа пользователя к ресурсам включает две процедуры: во-первых, пользователь должен доказать свою легальность (аутентификация), во-вторых, он должен получить разрешение на выполнение определенных операций с определенным ресурсом (авторизация). В системе Kerberos пользователь один раз аутентифицируется во время логического входа в сеть, а затем проходит процедуры аутентификации и авторизации всякий раз, когда ему требуется доступ к новому ресурсному серверу.

Выполняя логический вход в сеть, пользователь, точнее, Kerberos-клиент, установленный на его компьютере, посылает серверу аутентификации AS идентификатор пользователя ID (рис. 27.23).

Вначале сервер аутентификации проверяет в базе данных, имеется ли в ней запись о пользователе с таким идентификатором. Затем, если такая запись существует, он извлекает из нее пароль пользователя p. Данный пароль потребуется для шифрования всей информации, которую направит сервер аутентификации Kerberos-клиенту в качестве ответа. А ответ состоит из квитанции T_{TGS} на доступ к Kerberos-серверу квитанций и ключа сеанса K_S. Под сеансом здесь понимается все время работы пользователя от момента логического входа в сеть до момента логического выхода. Ключ сеанса потребуется для шифрования в процедурах аутентификации в течение всего пользовательского сеанса. Квитанция шифруется с помощью секретного мастер-ключа K, который разделяют серверы аутентификации и квитанций Kerberos KDC. Все вместе — зашифрованная квитанция и ключ сеанса — еще раз шифруется с помощью хеша пользовательского пароля p. Таким образом, квитанция шифруется дважды ключом K и паролем p. В приведенных обозначениях сообщение-ответ, которое сервер аутентификации посылает клиенту, выглядит так: $\{\{T_{TGS}\}K, K_S\}p$.

После того как такое ответное сообщение поступает на клиентскую машину, клиентская программа Kerberos просит пользователя ввести свой пароль. Когда пользователь вводит пароль, то Kerberos-клиент пробует с помощью хеша пароля расшифровать поступившее сообщение. Если пароль верен, то из сообщения извлекаются квитанция на доступ к серверу квитанций $\{T_{TGS}\}K$ (в зашифрованном виде) и ключ сеанса K_S (в открытом виде). Успешное дешифрование сообщения означает успешную аутентификацию. Заметим, что сервер аутентификации AS аутентифицирует пользователя без передачи пароля по сети. Нужно отметить, что успешность дешифрования будет проверена позже, когда пользователь попытается применить полученную квитанцию и ключ сеанса при обращении к серверу квитанций TGS.

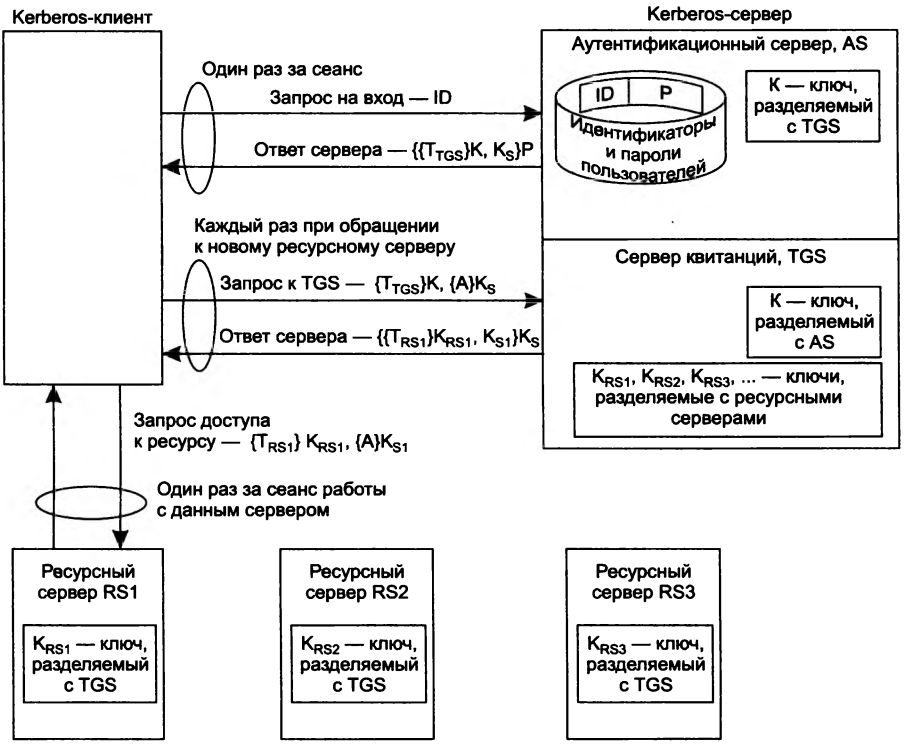


Рис. 27.23. Последовательность обмена сообщениями в системе Kerberos

Квитанция T_{TGS} на доступ к серверу квитанций TGS является удостоверением легальности пользователя и разрешением ему продолжать процесс получения доступа к ресурсу. Эта квитанция содержит:

- идентификатор пользователя;
- идентификатор сервера квитанций, на доступ к которому получена квитанция;
- отметку о текущем времени;
- период времени, в течение которого может продолжаться сеанс;
- копию ключа сеанса K_S.

Как уже отмечалось, клиент обладает квитанцией в зашифрованном виде. Шифрование повышает уверенность в том, что никто, даже сам клиент — обладатель данной квитанции, — не сможет квитанцию подделать, подменить или изменить. Только TGS-сервер, получив от клиента квитанцию, сможет ее расшифровать, так как в его распоряжении имеется ключ шифрования K.

Время действия квитанции ограничено длительностью сеанса. Разрешенная длительность сеанса пользователя, содержащаяся в квитанции на доступ к серверу квитанций, задается администратором и может изменяться в зависимости от требований к защищенности сети. В сетях с жесткими требованиями к безопасности время сеанса может быть ограничено 30 минутами, в других условиях это время может составить 8 часов. Информация, содержащаяся в квитанции, определяет ее срок годности. Предоставление квитанции на вполне

определенное время защищает ее от неавторизованного пользователя, который мог бы ее перехватить и применить в будущем.

Получение разрешения на доступ к ресурсному серверу

Итак, следующим этапом для пользователя является получение разрешения на доступ к ресурсному серверу (например, к файловому серверу или серверу приложений). Но для этого надо обратиться к TGS-серверу, который выдает такие разрешения (квитанции). Чтобы получить доступ к серверу квитанций, пользователь уже обзавелся квитанцией $\{T_{TGS}\}K$, выданной ему AS-сервером. Несмотря на защиту паролем и шифрование, пользователю, помимо квитанции, нужно кое-что еще, чтобы доказать серверу квитанций, что он имеет право на доступ к ресурсам сети.

Как уже упоминалось, первое сообщение от сервера аутентификации содержит не только квитанцию, но и секретный ключ сеанса K_S, который разделяется с сервером квитанций (TGS). Клиент задействует этот ключ для шифрования еще одной электронной формы, называемой *аутентификатором* $\{A\}K_S$. Аутентификатор A содержит идентификатор и сетевой адрес пользователя, а также собственную временну́ю отметку. В отличие от квитанции $\{T_{TGS}\}K$, которая в течение сеанса требуется многократно, аутентификатор предназначен для одноразового применения и имеет очень короткое время жизни — обычно несколько минут. Kerberos-клиент посылает серверу квитанций сообщение-запрос, содержащее квитанцию и аутентификатор: $\{T_{TGS}\}K$, $\{A\}K_S$.

Сервер квитанций расшифровывает квитанцию имеющимся у него ключом K, проверяет, не истек ли срок действия квитанции, и извлекает из нее идентификатор пользователя.

Затем TGS-сервер расшифровывает аутентификатор, применяя ключ сеанса пользователя K_S, который он извлек из квитанции. Сервер квитанций сравнивает идентификатор пользователя и его сетевой адрес с аналогичными параметрами в квитанции и сообщении. Если они совпадают, сервер квитанций удостоверяется, что данная квитанция действительно представлена ее законным владельцем. Применение ключа сеанса из зашифрованной квитанции говорит TGS-серверу, что квитанция действительно была выдана AS-сервером, так как она была расшифрована мастер-ключом, который известен только паре AS-TGS.

Заметим, что простое обладание квитанцией на получение доступа к серверу квитанций не доказывает идентичности пользователя. Так как аутентификатор действителен только в течение короткого промежутка времени, то маловероятно украсть одновременно и квитанцию, и аутентификатор и применить их в течение этого времени.

Каждый раз, когда пользователь обращается к серверу квитанций для получения новой квитанции на доступ к ресурсу, он посылает многоразовую квитанцию и новый аутентификатор.

Клиент обращается к серверу квитанций за разрешением на доступ к ресурсному серверу, который здесь обозначен как RS1. Сервер квитанций, удостоверившись в легальности запроса и личности пользователя, отсылает ему ответ, содержащий две электронные формы: многоразовую квитанцию T_{RS1} на получение доступа к запрашиваемому ресурсному серверу и новый ключ сеанса K_{S1}.

Квитанция на получение доступа шифруется секретным ключом K_{RS1}, общим только для сервера квитанций и того сервера, к которому предоставляется доступ, в данном случае — RS1. Сервер квитанций разделяет уникальные секретные ключи с каждым сервером сети. Эти ключи распределяются между серверами сети физическим способом или каким-либо

иным секретным способом при установке системы Kerberos. Когда сервер квитанций передает квитанцию на доступ к какому-либо ресурсному серверу, то он шифрует ее, так что только этот сервер сможет расшифровать ее с помощью своего уникального ключа.

Новый ключ сеанса K_{S1} содержится не только в самом сообщении, посылаемом клиенту, но и внутри квитанции T_{RS1}. Все сообщение шифруется старым ключом сеанса клиента K_S, так что его может прочитать только этот клиент. Учитывая введенные обозначения, ответ TGS-сервера клиенту можно представить в следующем виде: $\{\{T_{RS1}\}K_{RS1}, K_{S1}\}K_S$.

Получение доступа к ресурсу

Когда клиент расшифровывает поступившее сообщение, то он отсылает серверу, к которому он хочет получить доступ, запрос, содержащий квитанцию на получение доступа и аутентификатор, зашифрованный новым ключом сеанса: $\{T_{RS1}\}K_{RS1}$, $\{A\}K_{S1}$.

Это сообщение обрабатывается аналогично тому, как обрабатывался запрос клиента TGS-сервером. Сначала расшифровывается квитанция ключом K_{RS1}, затем извлекается ключ сеанса K_{S1} и расшифровывается аутентификатор. Далее сравниваются данные о пользователе, содержащиеся в квитанции и аутентификаторе. Если проверка проходит успешно, то доступ к сетевому ресурсу разрешается.

На этом этапе клиент тоже может захотеть проверить аутентичность сервера перед тем, как начать с ним работать. *Взаимная процедура аутентификации* предотвращает любую возможность попытки получения неавторизованным пользователем доступа к секретной информации от клиента путем подмены сервера.

Аутентификация ресурсного сервера в системе Kerberos выполняется в соответствии со следующей процедурой. Клиент обращается к серверу с предложением, чтобы тот прислал ему сообщение, в котором повторил временну́ю отметку из аутентификатора клиента, увеличенную на единицу. Кроме того, требуется, чтобы данное сообщение было зашифровано ключом сеанса K_{S1}. Чтобы выполнить такой запрос клиента, сервер извлекает копию ключа сеанса из квитанции на доступ, применяет этот ключ для расшифровки аутентификатора, наращивает значение временно́й отметки на единицу, заново зашифровывает сообщение с помощью ключа сеанса и возвращает сообщение клиенту. Клиент расшифровывает это сообщение, чтобы получить увеличенную на единицу временну́ю отметку.

При успешном завершении описанного процесса клиент и сервер удостоверяются в секретности своих транзакций. Кроме того, они получают ключ сеанса, который могут использовать для шифрования будущих сообщений.

Достоинства и недостатки

Изучая довольно сложный механизм системы Kerberos, нельзя не задаться вопросом: какое влияние оказывают все эти многочисленные процедуры шифрования и обмена ключами на производительность сети, какую часть ресурсов сети они потребляют, как это сказывается на ее пропускной способности?

Ответ весьма оптимистичный: если система Kerberos реализована и сконфигурирована правильно, ее работа сказывается на производительности сети незначительно. Так как квитанции используются многократно, сетевые ресурсы, затрачиваемые на запросы предоставления квитанций, невелики. Хотя передача квитанции при аутентификации логического входа несколько снижает пропускную способность, такой обмен требуется

в любых других системах и для любых методов аутентификации. Дополнительные же издержки незначительны. Опыт внедрения системы Kerberos показал, что время отклика при установленной системе Kerberos существенно не отличается от времени отклика без нее — даже в очень больших сетях с десятками тысяч узлов. Такая эффективность делает систему Kerberos весьма перспективной.

Среди уязвимых мест системы Kerberos можно назвать централизованное хранение всех секретных ключей. Успешная атака на Kerberos-сервер, в котором сосредоточена вся информация, критическая для системы безопасности, приводит к краху информационной защиты всей сети. Поэтому хранилище мастер-ключей должно быть хорошо защищено.

Для защиты секретных мастер-ключей в процессе первоначального создания (заведения новых учетных записей пользователей и ресурсных серверов в хранилище KDC) необходимо создавать защищенный канал между компьютером пользователя или ресурсным сервером и Kerberos KDC.

Возможен также полный отказ от паролей пользователя за счет цифровых сертификатов пользователя на первом этапе аутентификации, когда пользователь обращается к AS-серверу квитанцией T_{TGS}. Это расширение протокола Kerberos описано в документе RFC 4556, Public Key Cryptography for Initial Authentication in Kerberos (PKINIT). Предполагается, что пользователь применяет для логического входа смарт-карту, на которой хранится цифровой сертификат, выпущенный доверенным сертификационным центром. После того как пользователь локально аутентифицирует себя как легальный владелец смарт-карты, его компьютер передает запрос на логический вход AS-серверу системы Kerberos, в котором содержится цифровой сертификат пользователя с его открытым ключом, а также стандартный для Kerberos аутентификатор (идентификатор пользователя, его сетевой адрес и временна́я отметка), зашифрованный закрытым ключом пользователя, хранящимся на смарт-карте.

AS-сервер проверяет подлинность сертификата путем обращения к серверу сертификатов (корпоративному или публичному), затем расшифровывает аутентификатор с помощью открытого ключа пользователя и извлекает из него идентификатор пользователя. Если этот идентификатор имеется в базе идентификаторов пользователей AS-сервера, то сервер отвечает стандартным образом, то есть посылает пользователю квитанцию T_{TGS}, зашифрованную ключом K, а также ключ сеанса K_S, — однако ответ шифруется открытым ключом пользователя, а не его паролем. Пользователь расшифровывает ответ с помощью своего закрытого ключа, и на этом работа расширения PKINIT заканчивается.

Еще одной слабостью системы Kerberos является то, что исходные коды приложений, доступ к которым осуществляется через Kerberos, должны быть соответствующим образом модифицированы. Такая модификация называется «керберизацией» приложения. Некоторые поставщики продают «керберизованные» версии своих приложений. Однако если такой версии нет и нет исходного текста, то Kerberos не может обслуживать доступ к такому приложению.

В заключение хочется еще раз подчеркнуть, что стандартная версия Kerberos (соответствующая документу RFC 4210 и некоторым другим документам RFC, разработанным в IETF рабочей группой Kerberos) выполняет только аутентификацию пользователей. Авторизация оставлена ресурсным серверам, которые должны, например, задействовать списки доступа при принятии решения о том, что может делать конкретный пользователь, представленный своим идентификатором в квитанции Kerberos, и что ему делать запрещено.

Выводы

Аутентификаторы пользователей разделяют на три класса, определяемые следующим образом: «что-то, что знаю», «что-то, что имею» и «что-то, чем являюсь».

Аутентификация может выполняться на основе многоразовых и одноразовых паролей, генерируемых программными и аппаратными аутентификаторами.

Аутентификация, в процессе которой используются методы шифрования, а аутентификатор не передается по сети, называется строгой аутентификацией. Примером строгой аутентификации может служить аутентификация по квитированию вызова, применяемая в протоколе PPP.

Для аутентификации документов используется электронная (цифровая) подпись. Цифровая подпись может быть получена на основе как симметричного, так и несимметричного шифрования.

Аутентификация с применением цифровых сертификатов является альтернативой применению паролей и представляется естественным решением для систем с большим числом пользователей.

Ограничения доступа могут быть представлены в виде: правил, конфигурационных параметров пользовательского интерфейса, матрицы доступа, списков управления доступом, списков разрешений.

Большинство реализуемых на практике методов управления доступом может быть отнесено к дискреционному, мандатному или ролевому методу.

Главные особенности дискреционного метода: права доступа описываются в виде списков ACL, право назначать права на доступ к объектам делегируется отдельным пользователям — владельцам объектов.

Главные особенности мандатного метода: авторизацию и управление доступом осуществляет центральный полномочный орган, решение о предоставлении права доступа принимается операционной системой динамически на основе простого правила.

Ролевая система управления доступом сочетает в себе черты мандатного и дискреционного методов управления доступом.

Основным методом аутентификации пользователей ОС является метод, основанный на применении многоразовых паролей, хотя в общем случае поддерживаются все распространенные методы аутентификации — на основе многоразовых и одноразовых паролей (аппаратных и программных), биометрических данных и цифровых сертификатов.

Целью процедур единого логического входа является создание такого порядка аутентификации, при котором пользователь выполняет вход в сеть только один раз, доказывая свою аутентичность с помощью любого способа аутентификации, а затем результат этой аутентификации прозрачным для пользователя способом учитывается каждый раз, когда ему нужно доказывать свою аутентичность какому-либо серверу или приложению. Примером системы, реализующей процедуру единого логического входа, является сетевая служба Kerberos.

Контрольные вопросы

1. Отметьте слабости аутентификации с использованием многоразовых паролей:
   а) необходимость передачи пароля по сети в открытом виде;

б) возможность подслушивания, подглядывания, «выманивания» пароля;

в) ручная синхронизация;

г) трудность запоминания надежных паролей;

д) сложность реализации.

2. Что из перечисленного содержится в сертификате? Варианты ответов:

а) информация о владельце сертификата;

б) информация о сертифицирующей организации;

в) открытый ключ владельца сертификата;

г) закрытый ключ владельца сертификата;

д) открытый ключ сертифицирующей организации;

е) закрытый ключ сертифицирующей организации.

3. Основная идея процедуры единого логического входа состоит в том, что:

а) пользователь выполняет логический вход в сеть только один раз при поступлении на работу, все остальное время система лишь обеспечивает его авторизацию;

б) все пользователи выполняют процедуру логического входа с одного и того же компьютера;

в) пользователь выполняет логический вход в сеть только один раз, затем результат этой аутентификации используется другими серверами и приложениями.

4. С какой целью в электронной подписи используется хеш-функция? Варианты ответов:

а) для уменьшения длины сообщения;

б) для обеспечения конфиденциальности;

в) для уменьшения времени получения электронной подписи.

5. Какие свойства из перечисленных характеризуют дискреционный метод управления доступом? Варианты ответов:

а) описание прав доступа дается в виде списков ACL;

б) право назначать права на доступ к объектам делегируются отдельным пользователям — владельцам объектов;

в) права доступа субъектов к объектам определяются правилом;

г) данная система управления доступом позволяет гарантированно проводить общую политику.

6. Какие из следующих утверждений справедливы? Варианты ответов:

а) работа системы Kerberos существенно снижает производительность сети;

б) на надежность системы Kerberos негативно влияет способ хранения секретных ключей;

в) слабостью системы Kerberos является необходимость модификации приложений, к которым осуществляется доступ;

г) Kerberos осуществляет все обмены данными с клиентами и серверами в зашифрованном виде с использованием асимметричного алгоритма шифрования.

ГЛАВА 28 Технологии безопасности на основе фильтрации и мониторинга трафика

Фильтрация

Под **фильтрацией трафика** понимается обработка IP-пакетов маршрутизаторами и файерволами, приводящая к отбрасыванию некоторых пакетов или изменению их маршрута.

Фильтрация трафика позволяет либо предотвратить атаку на сеть, заранее блокируя доступ к ней для некоторых внешних сетей и хостов, либо, если источник атаки не был предварительно заблокирован, остановить ее.

Условия фильтрации бывают самыми разными, и не всегда удается найти простой признак, по которому одни пакеты нужно пропускать, а другие — отбрасывать. К тому же такое условие почти всегда является компромиссом между предотвращением атаки и поддержанием должной функциональности защищаемого узла — чем больше потенциальных атак мы предотвращаем, тем больше урезаем функции узла TCP/IP, связанные с его обычной работой.

Поэтому фильтрацию можно рассматривать как инструмент, которым надо уметь пользоваться.

Виды фильтрации

Выборочная передача кадров/пакетов маршрутизатором осуществляется на основе стандартных и дополнительных правил, называемых также **фильтрами**.

Стандартные правила фильтрации присущи не только маршрутизаторам, но и другим коммуникационным устройствам — концентраторам и коммутаторам. Эти правила определяются функциональностью устройств. Так, стандартное (функциональное) правило фильтрации для *концентратора* заключается в том, что кадр, поступивший на любой его интерфейс, независимо от адреса назначения кадра *повторяется* на всех остальных его интерфейсах. *Коммутатор* же функционирует в соответствии с правилом, когда кадр, имеющий некоторый адрес назначения, повторяется только на том интерфейсе, к которому подключена подсеть, имеющая в своем составе узел с данным адресом. Что касается стандартных правил фильтрации для *маршрутизатора*, то они состоят в том, что пакет, поступивший на входной интерфейс, перемещается на тот или иной интерфейс (или отбрасывается) *на основе адресной таблицы* маршрутизатора, в которой учитываются параметры маршрутов и пакетов.

Дополнительные правила фильтрации, или **пользовательские фильтры**, задаются сетевыми администраторами исходя из политики безопасности или с целью изменения

стандартных маршрутов. Дополнительные правила фильтрации маршрутизаторов[1] могут учитывать:

- IP-адреса источника и приемника;
- MAC-адреса источника и приемника;
- идентификаторы интерфейсов, с которых поступают пакеты;
- типы протоколов, сообщения которых несут IP-пакеты (то есть TCP, UDP, ICMP или OSPF);
- номера портов TCP/UDP (то есть типы протоколов прикладного уровня).

При наличии пользовательского фильтра маршрутизатор сначала сравнивает описываемые этим фильтром условия с признаками пакета и при положительной проверке выполняет над пакетом ряд нестандартных действий. Например, пакет может быть отброшен (drop); направлен следующему маршрутизатору, отличающемуся от того, который указан в таблице маршрутизации; помечен как вероятный кандидат на отбрасывание при возникновении перегрузки. Одним из таких действий может быть и обычная передача пакета в соответствии с записями таблицы маршрутизации.

Фильтрация может преследовать разные цели, в том числе логическую структуризацию сети, нестандартную маршрутизацию, защиту сети от вредительского трафика.

Логическая структуризация, то есть разделение компьютерной сети на подсети и сегменты, самым непосредственным образом влияет на ее эффективность. Инструментом логической структуризации является фильтрация *пользовательского трафика*, проходящего через интерфейсы сетевых устройств на основе стандартных правил.

Нестандартная маршрутизация реализуется за счет фильтрации *маршрутных объявлений*. Протоколы IP-маршрутизации создают таблицы маршрутизации, на основе которых любой узел составной сети может обмениваться информацией с любым другим узлом. Благодаря этому принципу дейтаграммных сетей каждый пользователь Интернета может получить доступ к любому публичному сайту. (Напомним, что в сетях, основанных на технике виртуальных каналов, действует другое правило: взаимодействие произвольных узлов невозможно без предварительной процедуры установления между ними виртуального канала.) Однако такая всеобщая достижимость узлов в IP-сетях не всегда отражает потребности их владельцев. Поэтому многие маршрутизаторы поддерживают фильтрацию объявлений протоколов маршрутизации, что позволяет дифференцированно управлять достижимостью узлов.

Фильтрация трафика в целях безопасности является важным средством **защиты от атак**. Функцию фильтрации поддерживают файерволы разного типа, в том числе файерволы на базе маршрутизаторов.

Иногда эффективная фильтрация требует анализа не одного, а некоторой последовательности пакетов. Например, для того чтобы распознать атаку TCP SYN (подробнее о ней читайте в следующей главе), недостаточно принимать во внимание только признаки одного пакета, взятого в отдельности от остальных. В этом случае мы не сможем отличить нормальный запрос на установление TCP-соединения от атаки — на это и рассчитывает злоумышленник. Признаком атаки TCP SYN является большое количество TCP-пакетов

[1] Фильтрация трафика маршрутизаторами аналогична по принципу действия фильтрации, выполняемой коммутаторами локальных сетей (см. главу 13).

SYN от некоторого источника без TCP-пакетов ACK от того же источника, поэтому для обнаружения этой атаки нужно запоминать и анализировать достаточно длинные последовательности пакетов. В этом случае говорят о **фильтрации с запоминанием состояния** (stateful) трафика. В том же случае, когда правила фильтрации учитывают только признаки отдельного пакета, говорят о **фильтрации без запоминания состояния** (statless) трафика.

И наконец, можно классифицировать фильтрацию *по уровню стека протоколов*, на котором анализируются заголовки и данные пакетов.

Стандартные и дополнительные правила фильтрации маршрутизаторов Cisco

Рассмотрим примеры пользовательских фильтров, написанных на командном языке маршрутизаторов Cisco. Эти фильтры, называемые **списками доступа** (access list)[1], являются очень распространенным средством ограничения пользовательского трафика в IP-маршрутизаторах.

Существует два типа списков доступа Cisco:

- **стандартный список доступа** (Standard) позволяет задавать условия фильтрации, учитывающие только IP-адрес источника;
- **расширенный список доступа** (Extended) позволяет использовать в условиях фильтрации IP-адреса источника и приемника, порты TCP и UDP источника и приемника, а также типы сообщений некоторых других протоколов, например ICMP.

Как стандартный, так и расширенный список доступа может состоять из нескольких условий, каждое из которых записывается в виде отдельной строки. Условия применяются к пакету в том порядке, в котором они перечисляются в списке доступа до первого совпадения (оставшиеся условия не проверяются).

Стандартный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } {адрес_источника [ метасимволы_
источника ] |any }
```

Служебные слова стандартного списка доступа:

- `access-list` — служебное слово, с которого начинается каждая запись;
- `deny` — запрет прохождения пакета, если условие выполняется;
- `permit` — разрешение прохождения пакета, если условие выполняется;
- `any` — служебное слово, которое говорит о том, что условие должно быть применено к любому значению адреса источника.

Числовые параметры стандартного списка доступа:

- *номер_списка_доступа* — всем условиям одного и того же списка доступа присваивается один и тот же номер из диапазона 1–99;
- *адрес_источника* — IP-адрес источника;

[1] Напомним, что термин «список доступа» употребляется также при описании механизмов контроля доступа в ОС, но там он имеет другой смысл.

- *метасимволы_источника* используются аналогично маске, которая накладывается на поля IP-адреса источника поступившего пакета и сравнивается с параметром *адрес_источника*.

Пример стандартного списка доступа:

```
access-list 1 deny 192.78.46.0 0.0.0.255
```

Здесь:

- 1 — номер списка доступа;
- deny — пакет, который удовлетворяет условию данного списка доступа, должен быть отброшен;
- 192.78.46.0 — адрес источника;
- 0.0.0.255 — метасимволы источника.

Этот фильтр запрещает передачу пакетов, у которых в старших трех байтах адреса источника имеется значение 192.78.46.0.

Список доступа может включать более одного условия. В этом случае он состоит из нескольких строк с ключевым словом `access-list` с одним и тем же номером. Так, если мы хотим разрешить прохождение через маршрутизатор пакетов хоста 192.78.46.12, запрещая передачу пакетов одному из хостов сети 192.78.46.0/24, то список доступа будет выглядеть следующим образом:

```
access-list 1 permit 192.78.46.12 0.0.0.0
access-list 1 deny 192.78.46.0 0.0.0.255
access-list 1 permit any
```

Этот список разрешает прохождение через маршрутизатор пакетов, отправляемых с хоста 192.78.46.12, и запрещает передачу пакетов, отправляемых любым другим хостом подсети 192.78.46.0/24.

Расширенный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } ключевое_слово_протокола

{адрес_источника метасимволы_источника [ операция порт_источника] | any }
{ адрес_приемника метасимволы_приемника [ операция порт_приемника ] | any }
```

Параметры:

- *номер_списка_доступа* — номер списка доступа из диапазона 100-199;
- *ключевое_слово_протокола* — `ip`, `tcp`, `udp` или `icmp`;
- *операция*: `eq`, `lt`, `gt` (позволяет задать порт, диапазон портов UDP/TCP или тип пакета ICMP).

Расширенный список дает возможность фильтровать пакеты определенных приложений на основе известных портов TCP/UDP их серверной части. Рассмотрим несколько примеров.

```
access-list 105 permit tcp any host 210.135.17.101 eq 21
```

Эта запись разрешает прием запросов от любого хоста, направленных FTP-серверу (TCP-порт 21) с адресом 210.135.17.101 (используется дополнительное служебное слово `host` вместо маски 0.0.0.0).

```
access-list 101 deny ICMP any 192.78.46.0 0.0.0.255 eq 8
```

Эта запись запрещает передачу эхо-запросов (ping-запросов) от любого хоста к хостам подсети 192.78.46.0/24.

```
access-list 105 permit tcp any eq 80 any gt 1023 established
```

Эта запись разрешает клиентам веб-службы (они всегда имеют порт TCP > 1023) получать ответы от любых веб-серверов (порт 80), с которыми у них уже установлено TCP-соединение (служебное слово `established` оговаривает это, маршрутизатор проверяет данный факт по наличию признака ACK в пакете).

Список доступа можно применять к любому интерфейсу маршрутизатора и в любом направлении: если список применяется с ключевым словом `in`, то он действует на входящие в интерфейс пакеты. В этом случае говорят, что выполняется **входная фильтрация** (ingress filtering).

Например, написанный нами список доступа 1 можно применить к некоторому интерфейсу для обработки входящего трафика, используя следующую команду:

```
access-group 1 in
```

Если же применить список доступа с ключевым словом `out`, то он будет воздействовать на пакеты, исходящие из интерфейса, в этом случае будет выполняться **выходная фильтрация** (egress filtering).

Для обеспечения подотчетности необходимо *протоколирование событий*, связанных с фильтрацией пакетов. Маршрутизаторы Cisco могут помещать сообщения об обработке пакетов, удовлетворяющих условию некоторой записи списка доступа, в системный журнал маршрутизатора syslog. По умолчанию такая опция для каждой записи списка доступа неактивна, это сделано для уменьшения нагрузки на маршрутизатор. Для активизации протоколирования необходимо добавить к записи ключевое слово `log`, например:

```
access-list 102 permit TCP any 21 any log
```

В заключение отметим, что приведенный здесь пример языка для списков доступа маршрутизаторов Cisco является хотя и фирменной, но достаточно типичной реализацией, которая хорошо иллюстрирует возможности применения маршрутизаторов как файерволов. Отсутствие фильтрации с запоминанием состояния связано со стремлением не создавать слишком большую нагрузку на маршрутизатор и «не отвлекать» его от основных обязанностей. Это ограничение является главным отличием маршрутизаторов от программных и программно-аппаратных файерволов.

(S) *Фильтрация маршрутных объявлений*

Файерволы

Функциональное назначение файервола

Файервол (межсетевой экран, или **брандмауэр)** — это комплекс программно-аппаратных средств, осуществляющий информационную защиту одной части компьютерной сети от другой путем анализа и фильтрации проходящего между ними трафика.

Файервол осуществляет экранирование защищаемого объекта и формирует его внешнее представление. Современные файерволы достигли очень высокого уровня защищенности, удобства использования и администрирования; в сетевой среде они являются первым и весьма мощным рубежом обороны.

ПРИМЕЧАНИЕ

Исходным значением термина «файервол» (от англ. firewall) является элемент конструкции дома, а именно стена, сделанная из огнеупорного материала и препятствующая распространению огня между частями дома (обычно принадлежащими разным собственникам). Термин «брандмауэр» (от нем. brandmauer) много лет назад пришел в русский язык из немецкого. Изначально он обозначал перегородку в поезде, отделяющую область топки паровоза от пассажирского отделения. Интересно, что немецкие специалисты в области безопасности для обозначения межсетевого экрана используют англоязычный термин firewall. В русском языке для термина «файервол» используются и другие транслитерации: файрволл, файрвол, фаервол.

Для того чтобы фильтровать трафик, файервол должен иметь по крайней мере два сетевых интерфейса: с внутренней сетью и с внешней сетью (рис. 28.1). Файервол защищает внутреннюю сеть (например, локальную сеть предприятия или, как вырожденный случай, отдельный компьютер пользователя) от угроз, исходящих из внешней сети (мы будем, как правило, подразумевать под такой сетью Интернет). Файервол может также защищать одну внутреннюю сеть предприятия от другой, если в соответствии с принципом минимума полномочий пользователям этих сетей не требуется полный взаимный доступ к ресурсам друг друга.

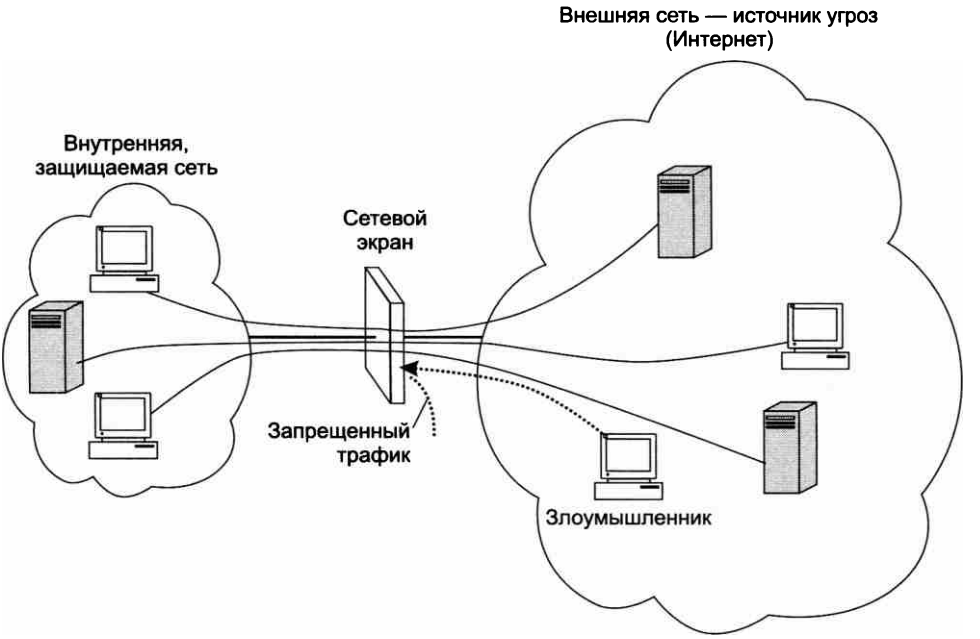


Рис. 28.1. Файервол защищает внутреннюю сеть от угроз, исходящих из внешней сети

Для эффективного выполнения файерволом его главной функции — анализа и фильтрации трафика — необходимо, чтобы через него проходил *весь* трафик, которым обмениваются узлы защищаемой части сети с узлами Интернета. В том случае, когда сеть связана с внешними сетями несколькими линиями связи, каждая линия связи должна быть защищена файерволом.

Файервол защищает сеть не только от несанкционированного доступа и атак внешних злоумышленников, но и от ошибочных действий пользователей защищаемой сети, например таких, как передача во внешнюю сеть конфиденциальной информации.

Основными функциями файервола являются:

- фильтрация трафика в целях защиты внутренних ресурсов сети;
- аудит — файервол должен фиксировать все события, связанные с обнаружением и блокировкой подозрительных пакетов.

Наряду с этими двумя базовыми функциями на файервол могут быть возложены и другие вспомогательные функции защиты, в частности:

- антивирусная защита;
- шифрование трафика;
- логическое посредничество между внутренними клиентами и внешними серверами (функция прокси-сервера);
- фильтрация сообщений по содержимому, включая типы передаваемых файлов, имена DNS и ключевые слова;
- предупреждение и обнаружение вторжений и сетевых атак;
- функции VPN;
- трансляция сетевых адресов (NAT).

Как можно заметить, большинство из перечисленных функций часто реализуются в виде отдельных продуктов или в составе систем защиты других типов. Так, функции пакетной фильтрации встроены практически во все маршрутизаторы, задача обнаружения вирусов решается множеством разнообразных программ, шифрование трафика — неотъемлемый элемент технологий защищенных каналов и т. д. и т. п. Прокси-серверы часто поставляются в виде приложений, более того, они сами иногда интегрируют в себе многие функции, свойственные сетевым экранам, такие, например, как фильтрация по содержимому (контенту) или трансляция сетевых адресов.

Отсюда возникают сложности при определении понятия «файервол». Например, довольно распространено мнение, что файервол — это пограничное устройство, выполняющее фильтрацию пакетов (то есть маршрутизатор), а прокси-сервер — это совершенно отличный от файервола инструмент защиты. Другие настаивают, что прокси-сервер является непременным и неотъемлемым атрибутом любого файервола. Третьи считают, что файерволом может быть названо только такое программное или аппаратное устройство, которое способно отслеживать состояние потока пакетов в рамках соединения. Мы же в этой книге будем придерживаться следующей точки зрения: файервол — это программно-аппаратный комплекс, выполняющий разнообразные функции по защите внутренней сети, набор которых может меняться в зависимости от типа, модели и конкретной конфигурации файервола, при этом минимальный набор функций должен включать фильтрацию трафика для предотвращения сетевых атак и аудит событий, связанных с фильтрацией.

Пример-аналогия

Функционально сетевой экран можно сравнить с системой безопасности современного аэропорта. Аналогии здесь достаточно очевидные (рис. 28.2): самолет соответствует защищаемой внутренней сети, а внешняя сеть, из которой приходит потенциально опасный трафик, — внешнему миру, откуда прибывают будущие пассажиры самолета, готовящегося к полету, при этом не все они приезжают с чистыми и ясными намерениями.

Рис. 28.2. Зона контроля аэропорта как аналогия сетевого экрана

В потоке пассажиров, постоянно входящих в здание аэропорта, могут встречаться различные злоумышленники. Наиболее зловещие — террористы — пытаются пронести на борт взрывчатку (в сетевом мире — пакеты, несущие во внутреннюю сеть вирусы, способные «взорвать» серверы и компьютеры пользователей) или оружие для захвата самолета в воздухе (атака по захвату управления удаленным компьютером). Контрабандисты несут с собой незадекларированные ценности (запрещенный контент), а некоторые личности пытаются попасть в самолет по поддельным документам (несанкционированный доступ к внутренним ресурсам сети).

Для того чтобы отфильтровать трафик пассажиров, система безопасности аэропорта пропускает всех пассажиров и их багаж через единственно возможный путь — зону контроля. Так же поступают при защите сети, направляя весь входящий трафик через сетевой экран. В зоне контроля аэропорта применяются разнообразные средства проверки пассажиров и их багажа. Аутентификация происходит путем сличения паспортов с компьютерной базой данных, а лиц пассажиров — с фотографиями в паспортах. Сюда же можно отнести просвечивание сумок и чемоданов, проход пассажиров через металлодетекторы, а при первом подозрении — вытряхивание всех вещей, дотошная ручная проверка сумок и личный досмотр

пассажиров. Между злоумышленниками и службой безопасности постоянно происходит состязание в коварстве, с одной стороны, и находчивости — с другой. Новые трюки вызывают появление новых способов проверки. Например, пронос взрывчатки в подошве ботинка породил не очень приятную обязательную процедуру прохождения металлоискателя без ботинок, а использование террористами флаконов для маскировки жидких компонентов бомбы лишило пассажиров возможности брать с собой в салон самолета шампуни и другие жидкости в больших объемах.

Сетевые экраны тоже пытаются использовать все возможные средства и методы для противостояния разнообразным угрозам. С помощью паролей и цифровых сертификатов они проверяют аутентичность внешних узлов, пытающихся установить соединения с внутренними; отслеживают логику обмена пакетами для того, чтобы отразить атаки, основанные на искажении этой логики; «просвечивают» содержимое электронных писем и загружаемых документов, пытаясь блокировать запрещенный контент; сканируют загружаемые программы, проверяя их на наличие известных вирусов. Так же как и в зоне контроля аэропорта, здесь постоянно идет соревнование между хакерами, все время изобретающими новые методы атак, и разработчиками сетевых экранов, старающихся эти атаки обнаружить и пресечь.

Типы файерволов

Файерволы можно классифицировать по самым разным критериям. Мы здесь остановимся на следующих, на наш взгляд, самых важных технических характеристиках файерволов: способе реализации, способе фильтрации и уровне модели OSI.

Реализация файервола так же многовариантна, как и его функциональность. В качестве аппаратной составляющей сетевого экрана может выступать маршрутизатор или комбинация маршрутизаторов, компьютер или комбинация компьютеров, комбинация маршрутизаторов и компьютеров, наконец, это может быть специализированное устройство. Таким же разнообразием отличается и программная составляющая сетевого экрана, имеющая гибкую структуру и включающая в себя различные модули, функции которых могут широко варьироваться.

В самом общем виде по способу реализации различают программный, аппаратный и программно-аппаратный файерволы:

- **программный файервол** реализован как программная система, работающая под управлением универсальной ОС, такой как Microsoft Windows, Linux или Mac OS (возможно, имеющая версии для нескольких универсальных ОС);
- **аппаратный файервол** реализован как набор дополнительных функций маршрутизатора, касающихся фильтрации (реже — Ethernet -коммутатора);
- **программно-аппаратный файервол** включает как программную систему, так и специализированный сервер, операционная система которого и аппаратура имеют конфигурацию и настройки, оптимизированные для работы файервола (чаще всего в качестве такой специализированной платформы используется универсальная ОС с набором специфических настроек, обеспечивающих максимальный уровень безопасности, а также сервер, сертифицированный для работы с программным обеспечением файервола).

В зависимости от способа фильтрации различаются файерволы без запоминания состояния и файерволы с запоминанием состояния:

- **файерволы без запоминания состояния** (stateless) выполняют фильтрацию на основе статических правил, при этом не отслеживаются состояния соединений (сеансов);

- **файерволы с запоминанием состояния** (stateful) принимают решения динамически с учетом текущего состояния сеанса и его предыстории.

Файерволы с запоминанием состояния для каждого сеанса, который удовлетворяет некоторым условиям, создают динамическую структуру данных в специальной таблице состояний файервола. После прихода очередного пакета контролируемого сеанса состояние сеанса корректируется и принимается решение о выполнении заданного действия с пакетом — пропускать или отбрасывать. Отслеживание для протоколов состояний сеансов требует больших объемов ресурсов, именно поэтому файерволы с запоминанием состояния создаются на программно-аппаратных платформах, имеющих большую оперативную память для хранения таблицы состояний сеансов и быстродействующие процессоры для обработки в реальном времени поступающих пакетов. Если же ресурсов такого файервола оказывается недостаточно, то он вместо пользы может принести вред, когда внутренние серверы оказываются недоступными не из-за атак на них, а из-за заторов трафика на интерфейсах файервола.

Теперь, когда мы обсудили такую особенность файерволов, как способность работать в режимах с запоминанием и без запоминания состояния, можно сказать, что именно отсутствие у маршрутизаторов режима фильтрации с запоминанием состояния является *наиболее существенным их отличием* от программных и программно-аппаратных файерволов. Это ограничение связано со стремлением не создавать слишком большую нагрузку на маршрутизатор, чтобы «не отвлекать» его от выполнения основных обязанностей. В то же время существуют и исключения из этого правила: например, модели Juniper SRX являются, с одной стороны, файерволами, поддерживающими фильтрацию с запоминанием состояния и работающими на всех уровнях, включая прикладной, а с другой стороны, они поддерживают все функции «нормального» маршрутизатора, а не только их усеченный набор, как это часто происходит с программно-аппаратными файерволами.

Одной из наиболее важных характеристик файервола является *уровень протокола модели OSI*, на котором он работает. По этому признаку различают файерволы сетевого, сеансового и прикладного уровней.

Если же файервол анализирует и фильтрует трафик на нескольких уровнях, то его относят к самому высокому из всех этих уровней.

Уровень протокола, на котором работает файервол, часто используют в качестве интегральной характеристики, поскольку с ней коррелируют другие признаки файервола. Например, файерволы, работающие на сеансовом и прикладном уровнях, чаще относятся к разряду файерволов с запоминанием состояния, а более простые файерволы сетевого уровня — без запоминания. Далее мы приводим типичные сочетания характеристик файерволов, упорядоченные по уровням.

К **файерволам канального уровня** могут быть условно отнесены управляемые коммутаторы, обладающие расширенным набором функций, в том числе возможностью фильтрации кадров канального уровня на основе задаваемых администратором списков доступа.

Файерволы сетевого уровня, называемые также **файерволами с фильтрацией пакетов** (packet filtering firewall), в полном соответствии со своим названием решают задачу фильтрации пакетов по IP-адресам (как источника, так и приемника), а также по значению поля протокола верхнего уровня — в пакет сетевого уровня могут быть вложены сообщения протоколов TCP, UDP, ICMP и др. Более того, несмотря на свое название, такие файерволы работают и на более высоком, транспортном уровне, то есть на уровне портов TCP

и UDP, но только на основе статических правил, при которых не отслеживаются состояния соединений, то есть в режиме без запоминания состояния. Поэтому с помощью файервола сетевого уровня можно заблокировать доступ к определенному приложению, запретив прохождение пакетов с определенными номерами портов TCP или UDP, но нельзя защитить сеть от искаженного сеанса TCP или HTTP, потому что это требует отслеживания последовательности шагов в сеансе, а значит, и запоминания состояния сеанса, чего файерволы сетевого уровня делать не умеют.

Этому типу файерволов соответствуют маршрутизаторы, поддерживающие пользовательские фильтры, а также программные персональные файерволы операционных систем. Опытный администратор может задать достаточно изощренные правила фильтрации, учитывающие многие требования, касающиеся защиты ресурсов внутренней сети, тем не менее этот тип сетевых экранов уступает по степени защиты другим типам. Преимуществами файерволов сетевого уровня являются простота, невысокая стоимость и минимальное влияние на производительность сети (то есть их дополнительная работа по фильтрации трафика не замедляет маршрутизацию пакетов между двумя сетями).

Файерволы сеансового уровня отслеживают состояния сеансов протоколов, другими словами, выполняют операцию запоминания состояния на уровнях ниже прикладного. Для того чтобы контролировать процесс установления соединения, файервол должен фиксировать для себя текущее состояние соединения, то есть запоминать, какое последнее сообщение отправил клиент и какое сообщение он ожидает получить.

Прежде всего, имеется в виду состояние сеанса протокола TCP, его начальная *трехшаговая процедура* установления соединения. Файервол проверяет, насколько соответствует последовательность обмена сообщениями контролируемому протоколу. То есть, например, если клиент посылает TCP-сообщение *SYN*, запрашивающее TCP-соединение, сервер должен отвечать TCP-сообщением *ACK SYN*, а не посылать в ответ, например, свой TCP-запрос *SYN*. После того как файервол установил допустимость TCP-соединения, он начинает работать простым передаточным звеном между клиентом и сервером. Таким образом, файервол сеансового уровня может защитить сеть от различных типов TCP-атак, в которых нарушается логика установления соединения.

Поддержка запоминания состояния сеансов протоколов позволяет этому типу файервола защищать сеть не только от атак на протокол TCP, но и от некоторых других видов атак, которые можно распознать и остановить, анализируя не отдельные пакеты, а их последовательность. Например, атаку Ping flood можно распознать по слишком маленькому интервалу между эхо-запросами от одного и того же источника, для чего устанавливается предельно допустимый минимальный интервал между эхо-запросами, а затем фиксируется время прихода очередного запроса. Если оно оказывается меньше предельного, то пакет отбрасывается. Таким образом, запоминание состояния сеансов может обобщаться и на протоколы, работающие *без установления соединения* (ICMP, UDP, DNS), а это означает, что в отличие от сетевых файерволов файерволы сеансового типа способны защитить сеть от некоторых видов DoS-атак, даже если они и не используют протокол TCP.

Файерволы прикладного уровня способны интерпретировать, анализировать и контролировать содержимое сообщений, которыми обмениваются приложения. Они также работают на основе фильтрации с запоминанием состояния, но анализируют состояния протоколов не только нижних уровней вплоть до транспортного, но и прикладного уровня, таких как протоколы SSH, HTTP, FTP, SQL, SMTP, POP3, IMAP, FTP, SSH, SQL и др.

ПРИМЕЧАНИЕ

Особым типом файерволов этого уровня является прокси-сервер, который перехватывает запросы клиентов к внешним серверам, с тем чтобы потом отправить их от своего имени. Этот тип сетевых экранов обеспечивает самый высокий уровень защиты, хотя и имеет свои недостатки, например требует больших вычислительных затрат. Кроме того, прокси-серверы могут скрывать адрес «доверившегося» ему клиента, что снижает эффективность других средств защиты. Работу прокси-серверов мы обсудим в следующем разделе.

Следует отличать функции по блокировке приложений, реализуемые файерволами сетевого уровня, от защиты приложений внутренней сети файерволами прикладного уровня. Файервол сетевого уровня понимает структуру заголовков пакетов TCP и UDP, за счет чего может запретить или разрешить прохождение пакетов с определенным номером программного порта TCP или UDP, а так как этот номер присвоен серверной части некоторого приложения, то блокируется весь трафик извне к этому приложению.

Файервол прикладного уровня действует более гибко. Он контролирует сеанс некоторого приложения и разрешает или запрещает определенные виды взаимодействия между внутренней и внешней частями этого приложения в соответствии с заданными правилами. Например, при контроле веб-службы файервол может разрешить использование только определенных команд протокола HTTP, а остальные запретить. В список запрещенных команд могут попасть опасные для веб-сервера команды PUT и DELETE. Аналогично, при контроле почтовой службы файервол прикладного уровня может не пропускать вовне письма, не подписанные цифровой подписью отправителя, если в этом состоит политика безопасности предприятия.

Файервол прикладного уровня, используемый в качестве корпоративного межсетевого экрана, чаще всего является интегрированным продуктом с модульной структурой, которая позволяет ему менять набор поддерживаемых функций фильтрации в зависимости от потребностей конкретной сети. За счет дополнительных модулей файерволы прикладного уровня могут поддерживать самые разные функции защиты программного обеспечения, например:

- антивирусный контроль загружаемых пользователем файлов и получаемых писем «на лету»;
- контроль контента, заключающийся, например, в ограничении доступа пользователей к внешним веб-сайтам, страницы которых содержат заданные ключевые слова, такой же контроль может применяться к электронным письмам, отправляемым вовне;
- транзитная аутентификация пользователей, обращающихся к некоторому приложению на внутреннем сервере, — эта функция полезна для тех приложений, которые либо не выполняют аутентификацию пользователей совсем, либо делают это незащищенным способом, как, например, FTP-сервер, который принимает пароли пользователей в открытом виде: файервол перехватывает обращение пользователя к FTP-серверу (команду USER) и организует сеанс логического входа пользователя, например, с сервером аутентификации Kerberos, если именно такой способ аутентификации применяется в корпоративной сети;
- централизованное шифрование электронных писем пользователей, что избавляет пользователей от необходимости конфигурировать такую функцию на своих клиентских компьютерах (для этого файервол должен хранить цифровые сертификаты пользователей);

- функции шлюза VPN с удаленными подразделениями предприятия и удаленными пользователями;
- трансляция внутренних IP-адресов пользователей на основе стандарта NAT.

Прокси-серверы

Функции прокси-сервера

Прокси-сервер (proxy server) — это особый тип приложения, которое выполняет функции посредника между клиентскими и серверными частями распределенных сетевых приложений, причем предполагается, что клиенты принадлежат внутренней (защищаемой) сети, а серверы — внешней (потенциально опасной) сети.

> Роль транзитного узла позволяет прокси-серверу логически разорвать прямое соединение между клиентом и сервером с целью контроля процесса обмена сообщениями между ними.

Подобно сетевому экрану, прокси-сервер может эффективно выполнять свои функции только при условии, что контролируемый им трафик не пойдет обходным путем.

Прокси-сервер может быть установлен не только на платформе, где работают все остальные модули файервола (рис. 28.3, *а*), но и на любом другом узле внутренней сети или сети демилитаризованной зоны (рис. 28.3, *б*). В последнем случае программное обеспечение клиента должно быть сконфигурировано таким образом, чтобы у него не было возможности установить прямое соединение с ресурсным сервером, минуя прокси-сервер.

Когда клиенту необходимо получить ресурс (файл, веб-страницу, почтовое сообщение) от какого-либо сервера, он посылает свой запрос прокси-серверу. Прокси-сервер анализирует этот запрос и на основании заданных ему администратором правил решает, каким образом он должен быть обработан (отброшен, передан без изменения ресурсному серверу, модифицирован тем или иным способом перед передачей, немедленно обработан силами самого прокси-сервера).

В качестве правил, которыми руководствуется прокси-сервер, могут выступать условия пакетной фильтрации. Правила могут быть достаточно сложными: например, в рабочие часы блокируется доступ к тем или иным узлам и/или приложениям, а доступ к другим узлам разрешается только определенным пользователям, причем для FTP-серверов пользователям разрешается делать лишь загрузку, а выгрузка запрещается. Прокси-серверы могут также фильтровать почтовые сообщения по типу пересылаемого файла (например, запретить получение приложений формата MP3) и по их контенту. К разным пользователям могут применяться разные правила фильтрации, поэтому часто на прокси-серверы возлагается задача аутентификации пользователей.

Если после всесторонней оценки запроса от приложения прокси-сервер констатирует, что запрос удовлетворяет условиям прохождения дальше во внешнюю сеть, то он по поручению приложения, но от своего имени выполняет процедуру соединения с сервером, затребованным данным приложением.

В некоторых случаях прокси-сервер может изменять запрос клиента. Например, если в него встроена функция трансляции сетевых адресов (см. далее раздел «Файерволы с функцией

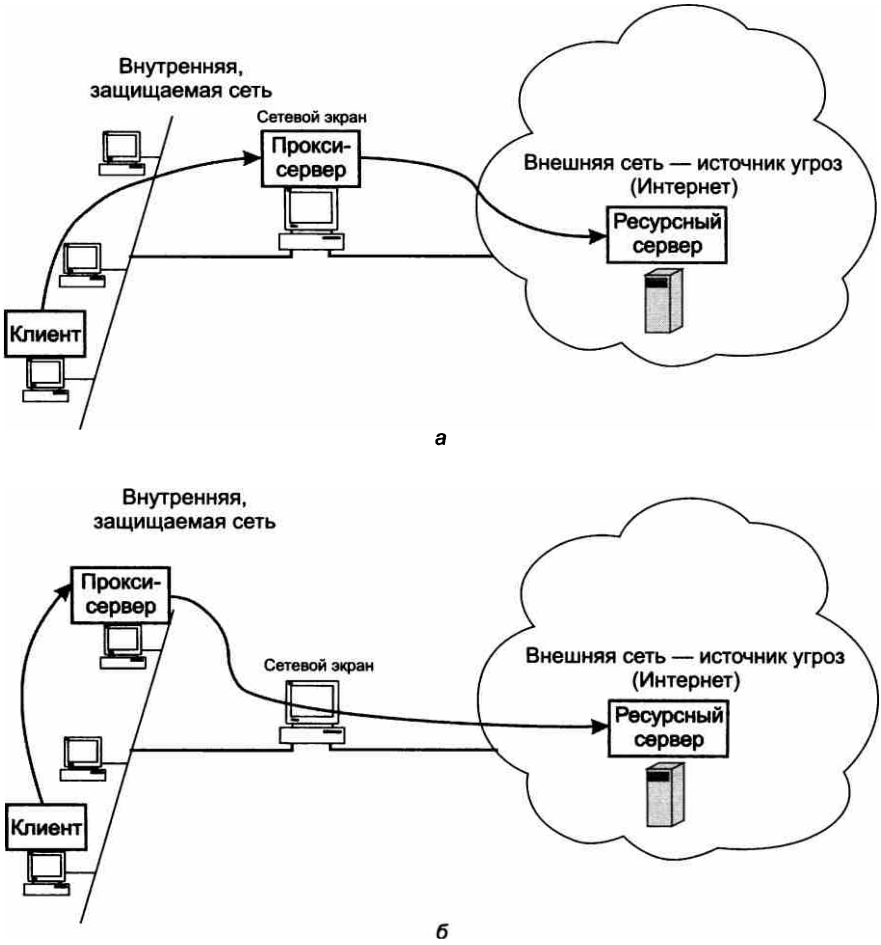


Рис. 28.3. Варианты размещения прокси-сервера

NAT»), он может подменять в пакете запроса IP-адреса и/или номера портов TCP и UDP отправителя. Таким способом прокси-сервер лишает злоумышленника возможности сканировать внутреннюю сеть для получения информации об адресах узлов и структуре сети. Единственный доступный злоумышленнику адрес в таком случае — это адрес компьютера, на котором выполняется программа прокси-сервера. Поэтому многие атаки, построенные на знании злоумышленником адресов узлов внутренней сети, становятся нереализуемыми.

Прокси-сервер, выступая посредником между клиентом и сервером, взаимодействующими по определенному протоколу, не может не учитывать специфику этого протокола. Так, для каждого из протоколов HTTP, HTTPS, SMTP/POP, FTP, telnet существует особый прокси-сервер, ориентированный на использование соответствующими приложениями: веб-браузером, программой электронной почты, FTP-клиентом, клиентом telnet. Каждый из этих посредников принимает и обрабатывает пакеты только того типа приложений,

для обслуживания которого он был создан. Обычно несколько разных прокси-серверов объединяют в один программный продукт.

Посмотрим, как учитывает специфику протокола *прокси-сервер, ориентированный на веб-службу*. Этот тип прокси-сервера может, например, выполнить собственными силами запрос веб-клиента, не отсылая его к соответствующему веб-серверу. Работая транзитным узлом при передаче сообщений между браузерами и веб-серверами Интернета, прокси-сервер не только передает клиентам запрашиваемые веб-страницы, но и сохраняет их в своей кэш-памяти на диске. В соответствии с алгоритмом кэширования на диске прокси-сервера оседают наиболее часто используемые веб-страницы. При получении запросов к веб-серверам прокси-сервер прежде всего проверяет, есть ли запрошенная страница в его кэше. Если есть, то она немедленно передается клиенту, а если нет, то прокси-сервер обычным образом делает запрос от имени своего доверителя. Прокси-сервер веб-службы может осуществлять административный контроль проходящего через него контента, в частности ограничивать доступ клиента к сайтам, имеющим IP-адреса или DNS-имена из «черных списков». Более того, он может фильтровать сообщения на основе ключевых слов.

Различают прокси-серверы прикладного и сеансового уровней.

Прокси-сервер прикладного уровня, как это следует из его названия, умеет «вклиниваться» в процедуру взаимодействия клиента и сервера по одному из прикладных протоколов, например HTTP, HTTPS, SMTP/POP, FTP или telnet. Чтобы выступать в роли посредника на прикладном уровне, прокси-сервер должен «понимать» смысл команд, «знать» форматы и последовательность сообщений, которыми обмениваются клиент и сервер соответствующей службы. Это дает возможность прокси-серверу проводить анализ содержимого сообщений и делать заключения о подозрительном характере того или иного сеанса.

Прокси-сервер сеансового уровня выполняет свою посредническую миссию на транспортном уровне, контролируя TCP-соединение. Очевидно, что, работая на более низком уровне, прокси-сервер обладает гораздо меньшим «интеллектом» и имеет меньше возможностей для выявления и предупреждения атак. Однако он обладает одним очень важным преимуществом перед прокси-сервером прикладного уровня — универсальностью, то есть он может быть использован любыми приложениями, работающими по протоколу TCP (а в некоторых случаях и UDP).

«Проксификация» приложений

Список приложений (точнее, их клиентских частей), которые должны передавать свои запросы во внешнюю сеть исключительно через прокси-сервер, определяется администратором. А чтобы эти приложения поддерживали такой режим выполнения, их программы должны быть соответствующим образом написаны.

Точнее, приложения должны быть оснащены средствами, которые распознавали бы запросы к внешним серверам и перед отправкой преобразовывали эти запросы так, чтобы все они попадали на соответствующий прокси-сервер, а не передавались в соответствии со стандартным протоколом прямо на сервер-адресат. Эти средства должны также поддерживать протокол обмена сообщениями приложения-клиента с прокси-сервером. В последние годы в большинстве приложений, ориентированных на работу через Интернет, предусмотрена встроенная поддержка прокси-сервера. Такой поддержкой, например, оснащены все веб-браузеры и все клиенты электронной почты, которыми мы сейчас пользуемся.

«Проксификация» приложения, изначально не рассчитанного на работу через прокси-сервер, требует изменения исходного кода с последующей перекомпиляцией — очевидно, что такая работа не представляет сложностей для разработчиков данного приложения, но администратор сети не всегда может ее выполнить, например из-за отсутствия исходного кода или же необходимой квалификации программиста. Задача администратора заключается в приобретении готовых приложений, совместимых с используемым в сети прокси-сервером. Однако даже приобретение готового «проксифицированного» клиента не делает его готовым к работе — необходимо еще конфигурирование, в частности, нужно сообщить клиенту адрес узла сети, на котором установлен соответствующий прокси-сервер.

Как можно было ожидать, процедура «проксификации» значительно упрощается для прокси-сервера сеансового уровня. Для «проксификации» приложения в этом случае достаточно внести простейшие исправления в исходный текст, которые сводятся к замене всех *стандартных вызовов сетевых функций* версиями этих функций из библиотеки процедур соответствующего прокси-сервера, а затем выполнить перекомпиляцию его программы.

Имеется еще один подход к «проксификации» — встраивание поддержки прокси-сервера в операционную систему. В этом случае приложения могут оставаться в полном «неведении» о существовании в сети прокси-сервера, за них все необходимые действия выполнит ОС.

Помимо основных функций, многие прокси-серверы могут выполнять другие полезные операции, например обнаруживать вирусы еще до того, как они попали во внутреннюю сеть, или собирать статистические данные о работе пользователей сети в Интернете.

Файерволы с функцией NAT

Одной из функций файервола является **трансляция сетевых адресов** (Network Address Translation, NAT). В этом случае фильтрация трафика заключается не в пропуске или отбрасывании пакетов, а в замене внешнего IP-адреса пакета, который использовался при маршрутизации пакета через Интернет, на внутренний, требуемый для маршрутизации во внутренней сети, корпоративной или персональной.

Сегодня существуют две причины обращения к технологии NAT: одна из них — дефицит адресов IPv4, другая — скрытие адресов хостов для повышения безопасности сети. В том и в другом случае внутренняя сеть использует *частные адреса*, которые заменяются одним или несколькими публичными адресами при отправке пакетов во внешние сети. Применение NAT позволяет скрыть адреса узлов своей сети, чтобы не дать возможности злоумышленникам составить представление о структуре и масштабах корпоративной сети, а также о структуре и интенсивности исходящего и входящего трафиков.

Технология NAT стояла у истоков зарождения файерволов как отдельного класса продуктов. В начале 90-х годов, когда дефицит адресов IPv4 еще мало ощущался, несколько специалистов основали компанию Network Translation и разработали программный продукт PIX, который позволял транслировать сетевые адреса. Позднее эту компанию приобрела компания Cisco, а программный продукт стал знаменитым фаейволом Cisco PIX Firewall, являющимся одним из флагманов средств защиты этого класса.

Традиционная технология NAT

Технология трансляции сетевых адресов имеет несколько разновидностей, наиболее популярная из которых — **традиционная технология трансляции сетевых адресов** — позволяет узлам из частной сети прозрачным для пользователей образом получать доступ к узлам внешних сетей. Подчеркнем, что в данном варианте NAT решается проблема организации только тех сеансов связи, которые *исходят* из частной сети. Направление сеанса в данном случае определяется положением инициатора: если обмен данными инициируется приложением, работающем на узле внутренней сети, то сеанс называется исходящим, несмотря на то что в его рамках в сеть могут поступать данные извне[1].

Идея технологии NAT состоит в следующем. Пусть сеть предприятия образует тупиковый домен, узлам которого присвоены частные адреса (рис. 28.4). На маршрутизаторе, связывающем сеть предприятия с внешней сетью, установлено программное обеспечение NAT. Это NAT-устройство динамически отображает набор частных адресов {IP*} на набор глобальных адресов {IP}, полученных предприятием от поставщика услуг и присвоенных внешнему интерфейсу маршрутизатора предприятия.

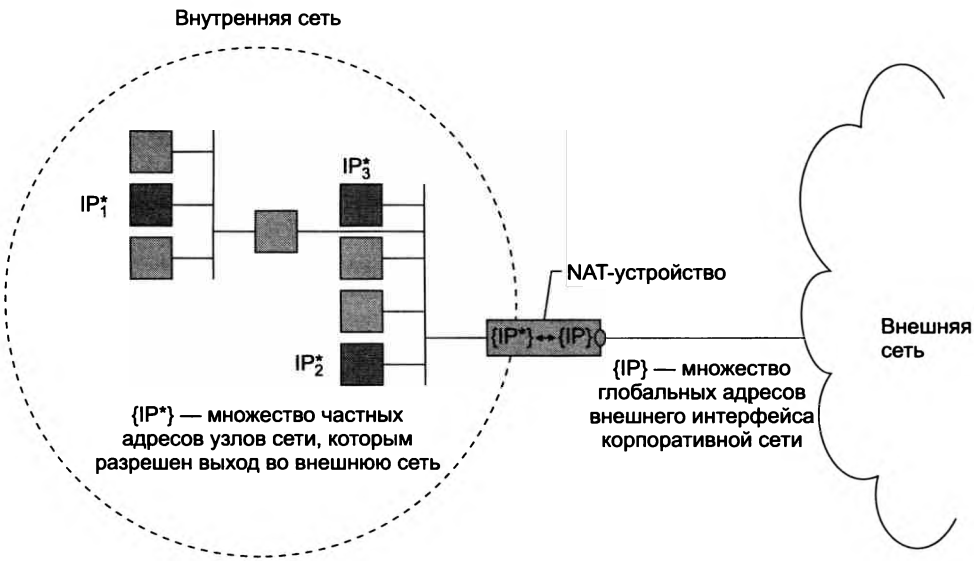


Рис. 28.4. Схема действия традиционной технологии NAT

Важным для работы NAT-устройства является правило распространения маршрутных объявлений через границы частных сетей. Объявления протоколов маршрутизации о внешних сетях «пропускаются» пограничными маршрутизаторами во внутренние сети и обрабатываются внутренними маршрутизаторами. Обратное утверждение неверно — маршрутизаторы внешних сетей не получают объявлений о внутренних сетях, объявления

[1] Традиционная технология NAT в виде исключения допускает сеансы обратного направления, заранее выполняя статическое взаимно однозначное отображение внутренних и внешних адресов для некоторого ограниченного набора узлов.

о них отфильтровываются при передаче на внешние интерфейсы. Поэтому внутренние маршрутизаторы «знают» маршруты ко всем внешним сетям, а внешним маршрутизаторам ничего не известно о существовании частных сетей.

Традиционная технология NAT подразделяется на технологии **базовой трансляции сетевых адресов** (Basic Network Address Translation, Basic NAT) и **трансляции сетевых адресов и портов** (Network Address Port Translation, NAPT). В технологии Basic NAT для отображения используются только IP-адреса, а в технологии NAPT — еще и так называемые транспортные идентификаторы, в качестве которых чаще всего выступают порты TCP и UDP.

Базовая трансляция сетевых адресов

Если количество локальных узлов, которым необходимо обеспечить выход во внешнюю сеть, меньше или равно имеющемуся количеству глобальных адресов, то для каждого частного адреса гарантировано однозначное отображение на глобальный адрес. В каждый момент времени количество внутренних узлов, которые получают возможность взаимодействовать с внешней сетью, ограничивается количеством адресов в глобальном наборе. Понятно, что в такой ситуации целью трансляции является не столько решение проблемы дефицита адресов, сколько обеспечение безопасности.

Частные адреса некоторых узлов могут отображаться на глобальные адреса *статически*. К таким узлам можно обращаться извне, используя закрепленные за ними глобальные адреса. Соответствие внутренних адресов внешним задается таблицей, поддерживаемой маршрутизатором или другим устройством (например, файерволом), на котором установлено программное обеспечение NAT.

В нескольких тупиковых доменах могут быть совпадающие частные адреса. Например, в сетях *A* и *B* на рис. 28.5 для внутренней адресации применяется один и тот же блок адресов 10.0.1.0/24. В то же время адреса внешних интерфейсов обеих сетей (181.230.25.1/24, 181.230.25.2/24 и 181.230.25.3/24 в сети *A* и 185.127.125.2/24 185.127.125.3/24 185.127.125.4/24 в сети *B*) уникальны глобально, то есть никакие другие узлы в составной сети их не используют. В данном примере в каждой из сетей только три узла имеют возможность «выхода» за пределы сети своего предприятия. Статическое соответствие частных адресов этих узлов глобальным адресам задано в таблицах пограничных устройств обеих сетей.

Когда узел 10.0.1.4 сети *A* посылает пакет хосту 10.0.1.2 сети *B*, то он помещает в заголовок пакета в качестве адреса назначения глобальный адрес 185.127.125.3/24. Узел-источник по умолчанию направляет пакет своему маршрутизатору R1, которому известен маршрут к сети 185.127.125.0/24. Маршрутизатор передает пакет на пограничный маршрутизатор R2, которому также известен маршрут к сети 185.127.125.0/24. Перед отправкой пакета модуль NAT, работающий на данном пограничном маршрутизаторе, используя таблицу отображения, заменяет в поле адреса источника частный адрес 10.0.1.4 соответствующим ему глобальным адресом 181.230.25.1/24. Когда пакет после путешествия по внешней сети поступает на внешний интерфейс NAT-устройства сети *B*, глобальный адрес назначения 185.127.125.3/24 преобразуется в частный адрес 10.0.1.2. Пакеты, передаваемые в обратном направлении, проходят аналогичную процедуру трансляции адресов.

Заметим, что в описанной операции не требуется участие узлов отправителя и получателя, то есть она прозрачна для пользователей.

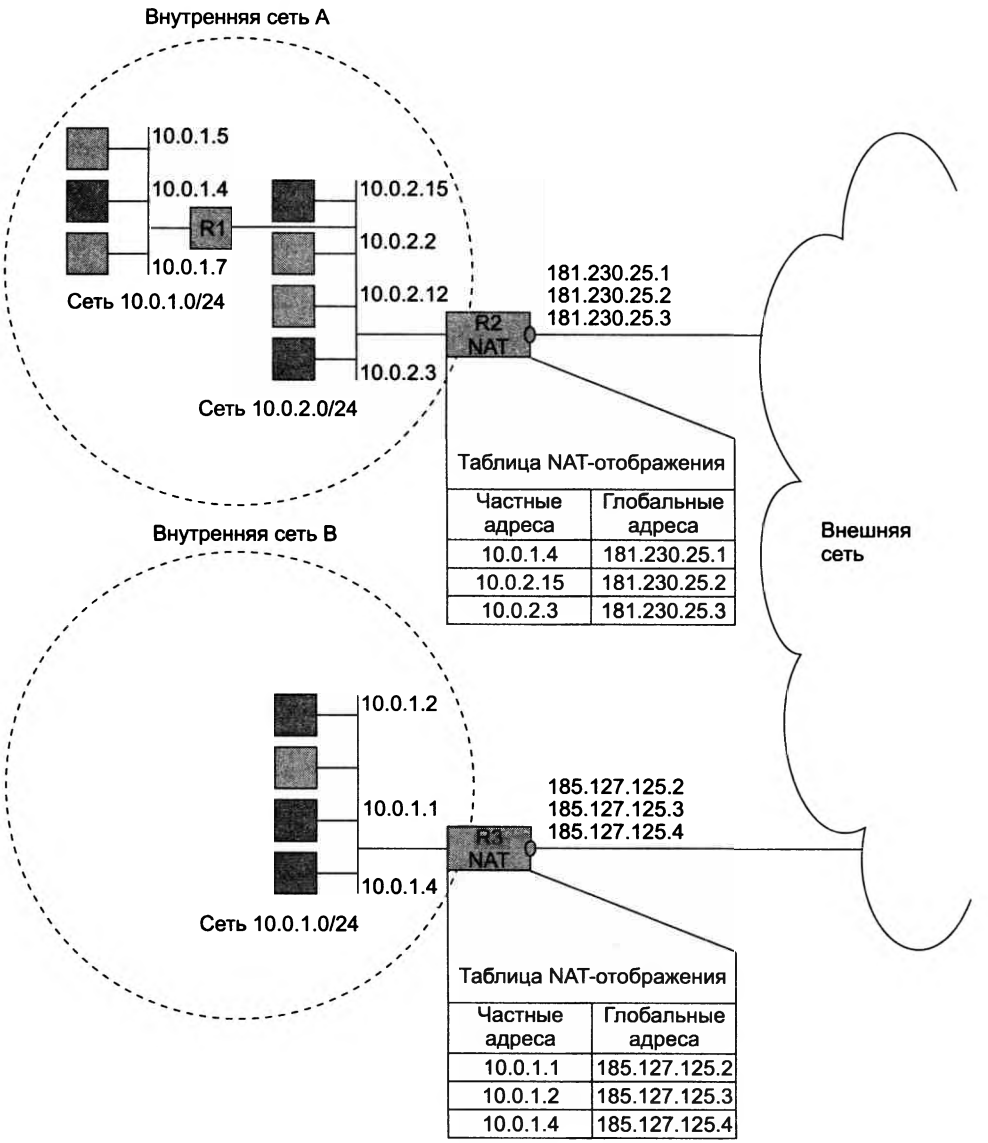


Рис. 28.5. Базовая трансляция сетевых адресов для исходящих сеансов

Трансляция сетевых адресов и портов

Пусть некоторая организация имеет частную IP-сеть и глобальную связь с поставщиком услуг Интернета. Внешнему интерфейсу пограничного маршрутизатора R2 назначен глобальный адрес, а остальным узлам сети организации назначены частные адреса. NAPT позволяет *всем* узлам внутренней сети одновременно взаимодействовать с внешними сетями, используя единственный зарегистрированный IP-адрес. Возникает законный вопрос: каким

образом внешние пакеты, поступающие *в ответ* на запросы из частной сети, находят узел-отправитель, ведь в поле адреса источника всех пакетов, отправляющихся во внешнюю сеть, помещается один и тот же адрес — адрес внешнего интерфейса пограничного маршрутизатора?

Для однозначной идентификации узла-отправителя привлекается дополнительная информация. Если в IP-пакете находятся данные протокола UDP или TCP, то в качестве такой информации выступает номер порта UDP или TCP соответственно. Но и это не вносит полной ясности, поскольку из внутренней сети может исходить несколько запросов с совпадающими номерами портов отправителя, а значит, опять возникает вопрос об однозначности отображения единственного глобального адреса на набор внутренних адресов. Решение состоит в том, что при прохождении пакета из внутренней во внешнюю сеть каждой паре {внутренний частный адрес; номер порта TCP или UDP отправителя} ставится в соответствие пара {глобальный IP-адрес внешнего интерфейса; назначенный номер порта TCP или UDP}. Назначенный номер порта выбирается произвольно, однако должно быть выполнено условие его уникальности в пределах всех узлов, получающих выход во внешнюю сеть. Соответствие фиксируется в таблице.

Эта модель при наличии единственного зарегистрированного IP-адреса, полученного от поставщика услуг, удовлетворяет требованиям по доступу к внешним сетям для большинства сетей средних размеров.

На рис. 28.6 приведен пример, когда в тупиковой сети *A* используются внутренние адреса из блока 10.0.0.0. Внешнему интерфейсу маршрутизатора этой сети поставщиком услуг назначен адрес 181.230.25.1.

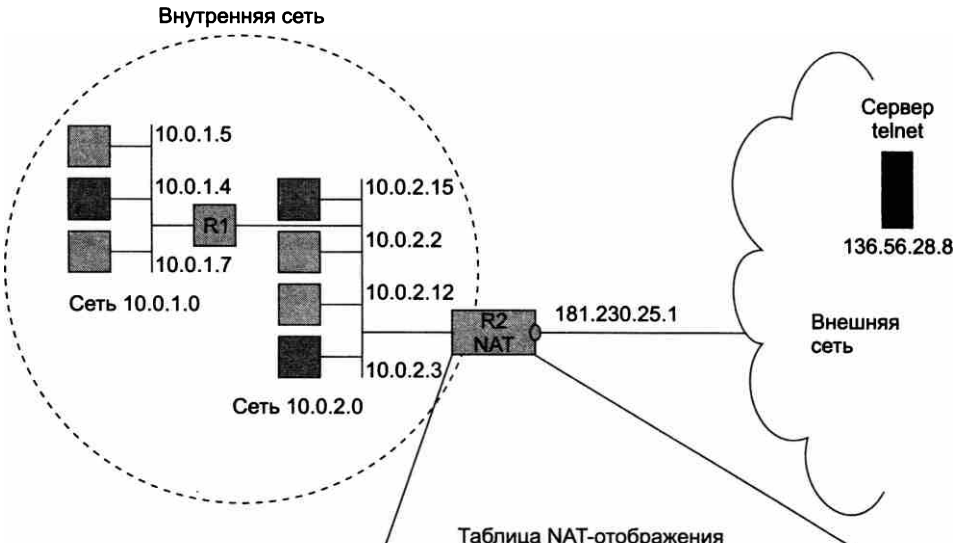


Таблица NAT-отображения

Частный адрес	Порт	Глобальный адрес	Назначенный порт
10.0.1.4	1245	181.230.25.1	3451
10.0.2.15	1245	181.230.25.1	3452
10.0.2.3	1045	181.230.25.1	3455

Рис. 28.6. Трансляция сетевых адресов и портов для исходящих сеансов TCP и UDP

Когда хост 10.0.1.4 внутренней сети посылает во внешнюю сеть пакет серверу telnet, то он в качестве адреса назначения использует его глобальный адрес 136.56.28.8. Пакет поступает маршрутизатору R1, который знает, что путь к сети 136.56.0.0/16 идет через пограничный маршрутизатор R2. Модуль NAPT маршрутизатора R2 транслирует адрес 10.0.1.4 и порт TCP 1245 источника в глобально-уникальный адрес 181.230.25.1 и уникально назначенный TCP-порт, в приведенном примере — 3451. В таком виде пакет отправляется во внешнюю сеть и достигает сервера telnet. Когда получатель генерирует ответное сообщение, то он в качестве адреса назначения указывает единственный зарегистрированный глобальный адрес внутренней сети, являющийся адресом внешнего интерфейса NAPT-устройства. В поле номера порта получателя сервер помещает назначенный номер TCP-порта, взятый из поля порта отправителя пришедшего пакета. При поступлении ответного пакета на NAPT-устройство внутренней сети именно по номеру порта в таблице трансляции выбирается нужная строка. По ней определяется внутренний IP-адрес соответствующего узла и действительный номер порта. Эта процедура трансляции полностью прозрачна для конечных узлов.

ПРИМЕЧАНИЕ

Обратите внимание: в таблице имеется еще одна запись с номером порта 1245, такая ситуация вполне возможна: операционные системы на разных компьютерах независимо присваивают номера портов клиентским программам. Именно для разрешения такой неоднозначности и привлекаются уникальные назначенные номера портов.

В технологии NAPT разрешаются только исходящие из частной сети сеансы TCP и UDP. Однако возникают ситуации, когда нужно обеспечить доступ к некоторому узлу внутренней сети извне. В простейшем случае, когда служба зарегистрирована, то есть ей присвоен хорошо известный номер порта (например, WWW или DNS), и, кроме того, эта служба представлена во внутренней сети в единственном экземпляре, задача решается достаточно просто. Служба и узел, на котором она работает, однозначно определяются хорошо известным зарегистрированным номером порта службы.

Завершая рассмотрение технологии NAT, заметим, что помимо традиционной технологии NAT существуют и другие ее варианты, например технология двойной трансляции сетевых адресов, когда модифицируются оба адреса — и источника, и приемника (в отличие от традиционной технологии NAT, когда модифицируется только один адрес). Двойная трансляция сетевых адресов необходима, когда частные и внешние адресные пространства имеют коллизии. Наиболее часто это происходит, когда внутренний домен имеет некорректно назначенные публичные адреса, которые принадлежат другой организации. Подобная ситуация может возникнуть из-за того, что сеть организации была изначально изолированной, и адреса назначались произвольно, причем из глобального пространства. Или же такая коллизия может быть следствием смены поставщика услуг, причем организация хотела бы сохранить старые адреса для узлов внутренней сети.

Программные файерволы хоста

Программные файерволы хоста являются частью его программного обеспечения, реализуя наряду с файерволом сети двухступенчатый контроль трафика. Программный файервол работает в режиме ядра ОС, контролируя сетевые интерфейсы хоста и перехватывая пакеты до передачи их протоколам стека TCP/IP.

Программные файерволы хоста являются, как правило, файерволами сетевого уровня без запоминания состояния сеанса. Отслеживание состояния сеанса требует значительных вычислительных ресурсов компьютера, поэтому поддержка файерволом хоста такой функции могла бы привести к существенному замедлению выполнения основных его функций.

Как и файерволы сетевого уровня на основе маршрутизаторов, программные файерволы хоста позволяют применять правила, учитывающие номера портов TCP/UDP. Это означает, что пользователь хоста может разрешать или запрещать доступ по сети к определенным приложениям хоста, пользующимся закрепленными за ними портами.

Рассмотрим, как можно блокировать доступ по сети к приложениям с помощью программного файервола iptables, имеющегося практически во всех версиях Unix/Linux. Этот файервол запускается как Unix-демон и работает на основе правил, записанных в текстовом виде в файле /etc/sysconfig/iptables. Правила состоят из трех секций:

- `INPUT` — правила фильтрации входящего трафика;
- `OUTPUT` — правила фильтрации исходящего трафика;
- `FORWARD` — правила фильтрации транзитного трафика в том случае, когда хост работает как IP-маршрутизатор, имея два сетевых интерфейса.

На рис. 28.7 показан пример правил секции INPUT. Как можно заметить, их синтаксис похож на синтаксис правил маршрутизаторов Cisco, которые мы рассмотрели в начале этой главы.

```
[root@ganymede sysconfig]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpts:vnc-server:5903
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpts:vnc-server:5903
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:oa-system
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:webcache
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpt:webcache
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:pcsync-https
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:msgsrvr
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ddi-tcp-1
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
```

Рис. 28.7. Правила секции INPUT файервола iptables

Рассмотрим, например, такое правило:

```
ACCEPT tcp - - anywhere anywhere state NEW tcp dpts:vnc-server:5903
```

Оно говорит о том, что пакеты, удовлетворяющие условию правила, должны быть приняты (`ACCEPT`). Этому правилу удовлетворяют пакеты протокола TCP (`tcp`) с любым адресом источника (`anywhere`) и любым адресом приемника (`anywhere`), относящиеся к новому сеансу

TCP (`state NEW`, то есть к пакетам с признаком SYN) и имеющие в поле порта назначения значения в диапазоне от 5900 (это стандартный порт сервиса vnc-server, поэтому порт задан с помощью своего имени) до 5903.

Список включает еще несколько аналогичных правил, а завершает его правило `REJECT all anywhere anywhere`, которое запрещает все, что не разрешено явно.

Типовые архитектуры сетей, защищаемых файерволами

Мы рассмотрели функциональные возможности файерволов по защите одной сети от возможных атак, исходящих от другой сети. В простейшем случае первая сеть — это единственная внутренняя сеть предприятия, а внешняя представлена всеми сетями Интернета, соединенными с внутренней сетью через единственную линию связи предприятия с провайдером Интернета. В реальности ситуация оказывается сложнее — сеть предприятия может состоять из нескольких сетей, при этом серверы и хосты этих сетей нуждаются в защите различного типа. Например, если в одной сети находится почтовый сервер и веб-сервер предприятия, а в другой — сервер базы данных клиентов предприятия, то доступ к ним должен регулироваться в соответствии с разными правилами.

Если добавить к этому, что многие предприятия соединяют свои сети с Интернетом несколькими линиями связи и, возможно, через несколько провайдеров, то защита сети предприятия приобретает еще одно измерение — защиту всего периметра сети с помощью нескольких файерволов, при этом их правила защиты должны быть согласованными. Под **сетью периметра** понимается совокупность всех связей корпоративной сети с внешними сетями — сетями провайдеров или корпоративными сетями других предприятий.

Для надежной и эффективной защиты корпоративной сети она должна быть *логически сегментирована* таким образом, чтобы ресурсы каждой подсети в отношении мер защиты были подобными. Ресурсы корпоративной сети, к которым обращаются внешние пользователи, безусловно, составляют в отношении мер безопасности отдельную группу. Это такие ресурсы, как почтовый сервер, веб-сервер, DNS-сервер. Повсеместной практикой является выделение таких ресурсов в отдельную группу и размещение их в подсети, которая получила название **демилитаризованной зоны**[1] (demilitarized zone, **DMZ**). В каком-то смысле зона DMZ подобна транзитной зоне аэропорта, потому что пассажирам разрешается использовать только ресурсы этой зоны, а доступ к внутренним ресурсам авиапредприятия для них закрыт.

Рассмотрим особенности организации защиты DMZ на примере сети, показанной на рис. 28.8. В этой сети на рубеже защиты установлено два маршрутизатора, между которыми располагается демилитаризованная зона. Маршрутизаторы здесь играют роль файерволов сетевого уровня. В данном случае сеть DMZ является также сетью периметра, так как только она соединяет внутреннюю сеть предприятия с внешними сетями.

---

[1] Иногда термины «сеть периметра» и «демилитаризованная зона» используются как синонимы.

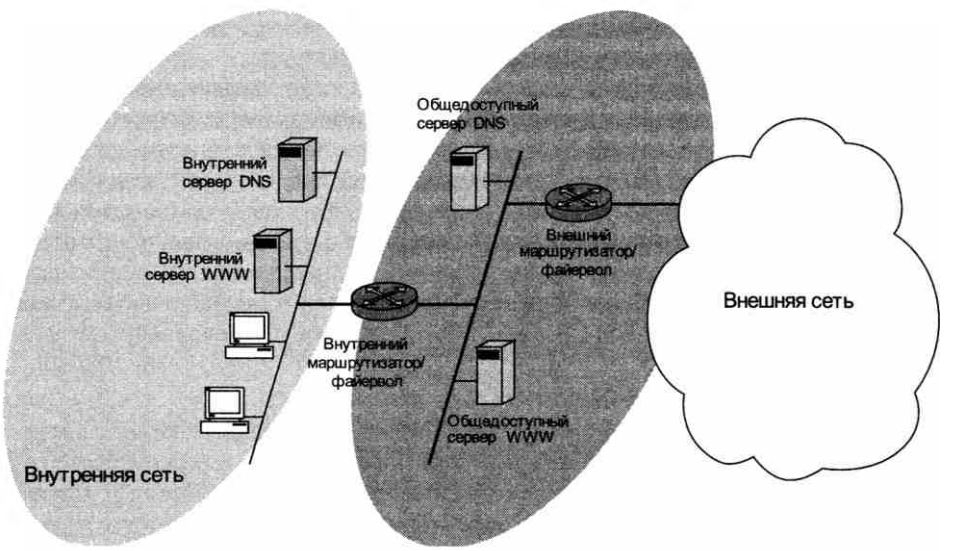


Рис. 28.8. Файервол на базе двух маршрутизаторов

В сети DMZ расположены два общедоступных сервера — внешний DNS-сервер и внешний веб-сервер предприятия. В этой зоне могут быть размещены также прокси-серверы. Учитывая, что само назначение этих компьютеров предполагает практически никак не ограничиваемый доступ к ним внешних пользователей (а значит, и злоумышленников), их необходимо защищать особенно тщательно. Главными задачами при защите этих компьютеров (называемых иногда **компьютерами-бастионами**) является обеспечение целостности и доступности размещенных на них данных для пользователей внешней сети. Эту задачу решают «индивидуальные» средства защиты, устанавливаемые на компьютерах-бастионах, такие, например, как антивирусные программы или фильтры спама. Кроме того, каждый сервер, к которому разрешено обращение внешних пользователей, должен быть сконфигурирован на поддержку только минимально необходимой функциональности. Например, публичный DNS-сервер предприятия не должен быть открытым для любых запросов, так как он может стать инструментом DDoS-атаки.

Чтобы пояснить, каким образом сеть периметра усиливает защиту внутренней сети, давайте посмотрим, что произойдет, если какой-либо злоумышленник сможет «взломать» первый рубеж защиты — внешний маршрутизатор — и начнет прослушивать трафик подключенной к нему сети периметра. Очевидно, что он получит доступ только к трафику общедоступных серверов, который не является секретным.

Внешний маршрутизатор призван фильтровать трафик с целью защиты сети периметра и внутренней сети. Однако строгая фильтрация в этом случае оказывается невостребованной. Общедоступные серверы по своей сути предназначены для практически неограниченного доступа. Что касается защиты внутренней сети, правила фильтрации для доступа к ее узлам и сервисам являются одними и теми же для обоих маршрутизаторов, поэтому внешний маршрутизатор может просто положиться в этом деле на внутренний маршрутизатор. Обычно внешний маршрутизатор находится в зоне ведения провайдера, и администраторы корпоративной сети ограничены в возможностях его оперативного

реконфигурирования. Это является еще одной причиной, по которой функциональная нагрузка на внешний маршрутизатор обычно невелика.

Основная работа по обеспечению безопасности локальной сети возлагается на внутренний маршрутизатор, который защищает ее как от внешней сети, так и от сети периметра. Правила, определенные для узлов сети периметра по доступу к ресурсам внутренней сети, часто бывают более строгими, чем правила, регламентирующие доступ к этим ресурсам внешних пользователей. Это делается для того, чтобы в случае взлома какого-либо компьютера-бастиона уменьшить число узлов и сервисов, которые впоследствии могут быть атакованы с этого компьютера. Именно поэтому внутренний маршрутизатор должен отбрасывать все пакеты, следующие во внутреннюю сеть из сети DMZ, исключая пакеты нескольких протоколов (например, HTTP, SMTP, DNS), абсолютно необходимых пользователям внутренней сети для обращения к внешним серверам, установленным в сети DMZ или же за пределами корпоративной сети.

Для исключения возможности обращения внешних пользователей к серверам внутренней сети можно разрешить пропуск к ней только тех TCP-пакетов, которые относятся к TCP-сеансам, установленным по инициативе внутренних пользователей. Например, для маршрутизаторов Cisco это можно сделать с помощью такой строки списка доступа:

```
access-list 200 permit tcp any 201.15.0.0 0.0.255.255 established
```

Здесь 201.15.0.0/16 — это диапазон адресов внутренней сети, а список доступа применяется во входном направлении к интерфейсу внутреннего маршрутизатора, к которому подключена сеть DMZ.

Защиту внутренней сети можно усилить, если в ней имеются аналоги внешних серверов, то есть в наше примере это веб-сервер и DNS-сервер. В такой конфигурации только этим северам в случае необходимости разрешается взаимодействовать с серверами зоны DMZ, внутренние же пользователи работают напрямую лишь с внутренними серверами. Например, DNS-сервером по умолчанию для пользователей сети должен быть назначен внутренний DNS-сервер, и только ему позволено изнутри обращаться к внешнему DNS-серверу в том случае, когда он не может разрешить запрос самостоятельно.

Защиту внутренних серверов можно усилить за счет использования частных IP-адресов во внутренней сети. В этом случае внутренний маршрутизатор при трансляции частных адресов должен поддерживать режим NAPT на своем интерфейсе, связывающем его с сетью DMZ.

Мониторинг трафика. Анализаторы протоколов

Файервол может успешно защитить внутреннюю сеть от разнообразных атак при условии, что его фильтры правильно сконфигурированы. Однако даже правильные фильтры конфигурируются статически, так что для подлинно эффективной защиты требуется заранее предвидеть все возможные атаки, а это в принципе невозможно. Любой новый тип атаки имеет все шансы «просочиться» через файервол и достичь внутренних серверов защищаемой сети.

Обнаружить следы атак, которые смогли преодолеть барьер файервола, можно путем мониторинга сетевого трафика.

Мониторинг сетевого трафика — непрерывный процесс инструментального автоматизированного наблюдения за отдельными параметрами трафика с целью проверки соблюдения SLA, планирования сети, а также предотвращения негативных событий, таких как технические аварии, угрозы и атаки злоумышленников.

Здесь мы рассмотрим следующие средства мониторинга сетевого трафика:

- *анализаторы протоколов*, или *сетевые сниферы*, позволяют захватывать трафик локальных сетей, представлять его в удобном для анализа виде, но собственно анализ данных оставляют администратору;
- *маршрутизаторы, поддерживающие протокол NetFlow*, собирают обобщенные данные о трафике глобальных сетей, передавая его для анализа программным системам NetFlow, которые автоматизируют поиск атак и угроз.
- *системы обнаружения вторжений* (Intrusion Detection Systems, IDS) специализируются на автоматическом распознавании вторжений и угроз в прослушиваемом трафике локальных сетей.

Анализаторы протоколов

Анализаторы протоколов способны на основе некоторых заданных оператором логических условий захватывать отдельные пакеты и декодировать их, то есть показывать в удобной для специалиста форме вложенность пакетов протоколов разных уровней друг в друга с расшифровкой содержания полей каждого пакета. Дружественный интерфейс, обычно присущий этому классу устройств, позволяет пользователю выводить результаты анализа интенсивности трафика; получать мгновенную и усредненную статистическую оценку производительности сети; задавать определенные события и критические ситуации для отслеживания их возникновения.

Анализатор протоколов представляет собой либо самостоятельное специализированное устройство, либо персональный компьютер, обычно переносной класса Notebook, оснащенный специальной сетевой картой и соответствующим программным обеспечением. Анализатор протоколов подключается к сети точно так же, как обычный узел. Отличие состоит в том, что анализатор протоколов может принимать все пакеты данных, передаваемые по сети, в то время как обычная станция — адресованные только ей. Программное обеспечение анализатора протоколов состоит из ядра, поддерживающего работу сетевого адаптера и декодирующего получаемые данные, и дополнительного программного кода, зависящего от типа исследуемой сети. В состав некоторых анализаторов может входить также *экспертная система*, способная выдавать пользователю рекомендации о том, какие эксперименты следует проводить в данной ситуации, что могут означать те или иные результаты измерений, как устранить некоторые виды неисправностей в сети.

Основываясь на результатах анализа содержимого пакетов того или иного протокола, можно оптимизировать производительность сети, находить и устранять неполадки, осуществлять обоснованное и взвешенное изменение каких-либо компонентов сети. Обычно для того, чтобы сделать какие-либо выводы о влиянии некоторого изменения на сеть, анализ протоколов выполняется и до, и после внесения этого изменения.

Возможности анализатора во многом определяются устройством и объемом *буфера захвата пакетов*. Буфер может располагаться на устанавливаемой сетевой карте, либо для него может быть отведено место в оперативной памяти одного из компьютеров сети.

Если буфер расположен на сетевой карте, то управление им осуществляется аппаратно, и за счет этого скорость ввода повышается. В случае недостаточной производительности процедуры захвата часть информации будет теряться, и анализ окажется невозможным. При заполнении буфера либо прекращается захват, либо заполнение начинается с начала буфера.

В качестве примера рассмотрим популярный свободно распространяемый программный анализатор протоколов **Wireshark** (прежнее название — *Ethereal*). Wireshark позволяет анализировать захваченный трафик, используя иерархическое представление полей пакетов (рис. 28.9).

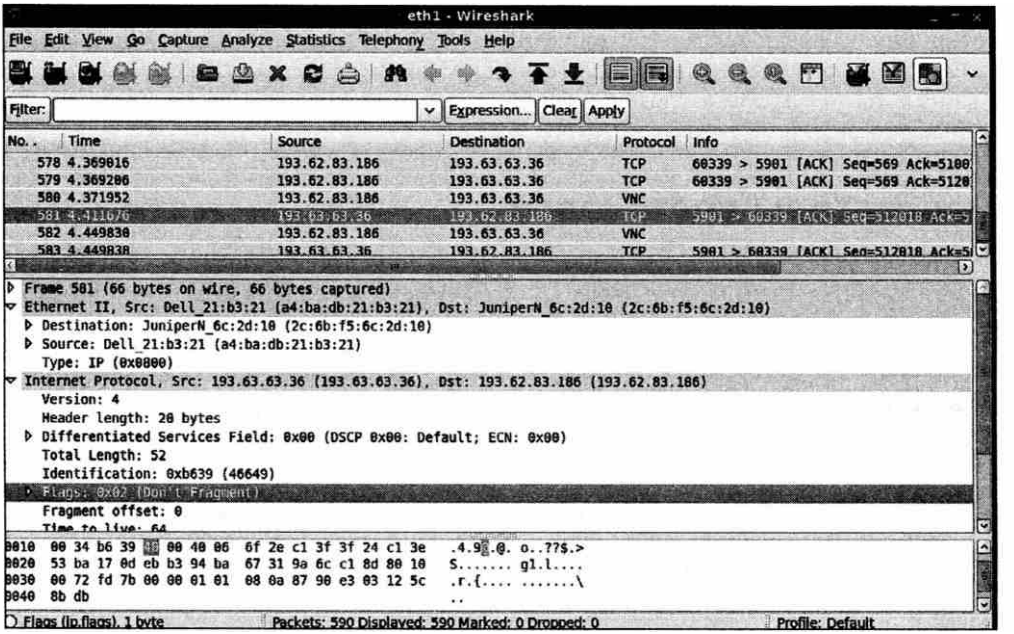


Рис. 28.9. Анализ трафика с помощью программы Wireshark

Верхняя панель окна результатов Wireshark укрупненно показывает основные параметры каждого захваченного пакета. Нижняя панель позволяет рассмотреть один из пакетов более детально, при этом те поля, которые состоят из нескольких подполей, можно раскрывать рекурсивно, добираясь до самого дна иерархии признаков пакета, например до признаков заголовка IP или TCP. Wireshark поддерживает весьма длинный список протоколов от канального до прикладного уровня — на практике это означает возможность раскрытия заголовков протоколов с пояснениями назначения каждого поля.

Wireshark дает возможность задавать фильтры двух типов: *фильтр захвата пакетов* и *фильтр отображения пакетов*; условия задания фильтров весьма гибкие, практически любое поле любого протокола может быть использовано в условиях этих фильтров. Захваченные кадры помещаются в файл с расширением .pcap (*p*acket *cap*ture), стандартизованный формат которого сегодня поддерживают практически все программные и программно-аппаратные средства мониторинга трафика.

Применение анализаторов протоколов для обнаружения атак требует значительного опыта, так как за десятками сеансов различных протоколов, часто несущих избыточную информацию, не так-то просто увидеть подозрительную активность. Поэтому часто анализ данных, собранных анализатором в файле формата PCAP, автоматизируют с помощью какой-нибудь из доступных программ анализа трафика[1]. В то же время предоставляемая анализаторами протоколов принципиальная возможность получить полную картину проходящего через сеть трафика дает шанс специалисту по безопасности разобраться в ситуации и обнаружить атаку даже в том случае, когда атака является совершенно новой и ее типичное поведение имеющемуся программному обеспечению анализа трафика пока не известно.

(S) *Анализатор протоколов Tcpdump*

Система мониторинга NetFlow

Система **NetFlow** сегодня является основным средством учета и анализа трафика, проходящего через маршрутизаторы и коммутаторы сети. Поддерживающие протокол NetFlow сетевые узлы не только выполняют свою основную работу — передачу пакетов в соответствии с адресом назначения, но и собирают статистику о проходящих через них потоках данных и периодически отправляют их в *коллекторы* для хранения и обработки такой информации.

Практически все ведущие производители сетевого оборудования поддерживают протокол NetFlow, так что для того, чтобы превратить ваш маршрутизатор в источник информации о проходящем трафике, достаточно активизировать на нем систему NetFlow и указать, куда нужно передавать статистику.

NetFlow собирает статистику не о каждом пакете, а о *потоке* пакетов, отсюда и название протокола (net — сеть, flow — поток). Под потоком понимается последовательность пакетов, принадлежащих одному и тому же соединению между определенными приложениями двух определенных компьютеров, например Skype-сеанс между двумя пользователями, передача файла с сервера на клиентский компьютер, чтение данных веб-страницы с сервера браузером клиентского компьютера. Аналогом потока можно считать данные телефонного разговора между двумя абонентами, однако между двумя компьютерами, в отличие от телефонов, может вестись сразу несколько «разговоров». Поэтому для определения потока нужно использовать не только адреса участвующих в сеансе компьютеров, но и дополнительные признаки. Собранную статистику о прохождении потоков можно использовать для различных целей, и одна из важнейших — *распознавание сетевых атак*.

NetFlow определяет поток как набор нескольких признаков, в число которых чаще всего входят[2]:

- IP-адрес источника;
- IP-адрес назначения;

---

[1] Не путать с анализатором протоколов.

[2] Такой набор параметров потока определен в NetFlow версии 5, являющейся на момент написания книги наиболее распространенной. В более развитых версиях, например в версии 9, определено более 50 параметров потока, имеется возможность собирать статистику о протоколах MPLS, BGP, IPv6.

- порт TCP/UDP источника;
- порт TCP/UDP приемника;
- тип протокола, переносимого IP-пакетом (полезно в тех случаях, когда это не TCP или UDP, например это может быть ICMP или OSPF);
- индекс интерфейса, на который получен пакет;
- качество обслуживания — значения байта ToS/DiffServ.

NetFlow собирает разнообразную статистику о потоке, такую как время начала и окончания потока, объем данных, переданных с момента начала потока, средняя скорость передачи данных, ну и, естественно, все параметры, определяющие поток, то есть адреса, порты и т. д. Кроме того, передается агрегированная информация о флагах заголовка TCP (SYN и др.).

Собранная статистика передается в коллекторы (один или несколько серверов) при окончании потока или же по истечении определенного периода времени, если поток еще не окончился.

Маршрутизатор может собирать данные NetFlow в двух режимах: непрерывном, когда обрабатывается каждый пакет, поступающий в маршрутизатор, и выборочный, когда обрабатывается только каждый n-й пакет. Выборочный режим менее надежен для распознавания атак, зато он создает гораздо меньше дополнительной нагрузки на маршрутизатор, а для магистрального маршрутизатора, через который проходят десятки, а иногда и сотни тысяч потоков, это существенно.

Типичная система мониторинга на базе NetFlow включает следующие функциональные компоненты (рис. 28.10):

- **Экспортер потока**, называемый также **сенсором**, агрегирует пакеты в потоки и передает статистические данные об этих потоках в один или несколько коллекторов. Экспортером чаще всего является маршрутизатор или коммутатор, хотя могут быть использованы и отдельно стоящие устройства, получающие данные путем зеркалирования порта коммутатора.
- **Коллектор** отвечает за прием, хранение и предварительную обработку данных о потоках, полученных от экспортера потока. Реализуется одним или несколькими серверами.
- **Программа-анализатор** анализирует полученные данные о потоках с целью распознавания возможных атак или возникновения перегрузок сети, установления состава и тенденций изменения трафика в сети.

Важно подчеркнуть, что, в отличие от анализаторов трафика и систем обнаружения атак, NetFlow собирает так называемые **метаданные** о трафике, не заглядывая в поля данных пакетов. Часто статистику NetFlow сравнивают с телефонным счетом, который показывает, с кем и сколько разговаривал данный абонент, но не раскрывает, о чем он говорил.

Однако знания метаданных часто бывает достаточно для того, чтобы распознать атаку. Для этого применяется общий принцип мониторинга сети — сравнение ее текущего поведения с «нормальным», то есть таким, которое устойчиво повторялось в прошлом и при этом мы знаем, что при этом атак в сети не наблюдалось.

Устойчивые значения статистических характеристик «нормального» поведения сети и ее узлов, которые получены на основании мониторинга сети на довольно значительном периоде времени (недели, месяцы), называются **базовым уровнем** (baseline) характеристик сети.

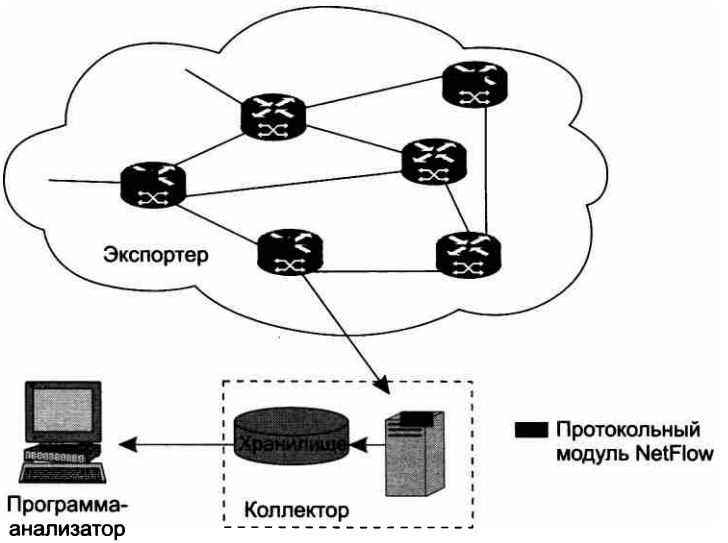


Рис. 28.10. Типичная система мониторинга на базе NetFlow

Другими словами, данные NetFlow служат для поиска аномалий в характере метаданных. Этот же прием используют службы безопасности банков: если вы обычно снимаете деньги в банкоматах Москвы, то снятие денег в Рейкьявике является для вас аномалией и ее нужно проверить — может быть, вы просто полетели посмотреть на гейзеры и водопады, а может, у вас украли данные вашей карточки.

Атака обычно генерирует не совсем обычный образец трафика, и существуют рекомендации для распознавания таких аномалий. Перечислим основные из них.

- *Выявление узлов с необычно большим числом запросов на установление соединений* (Top N Sessions). Если какой-либо узел вдруг вошел в число *N* узлов, наиболее активных в отношении установления сеансов, то это должно вызывать подозрения (значение *N* обычно выбирается не очень большим, к примеру 10). Такая активность характерна для DOS/DDOS-атак, узлов, зараженных червями, сканирования портов и некоторых других видов злоумышленной деятельности. Так, компьютер, зараженный червем, обычно пытается заразить таким кодом как можно больше других компьютеров и поэтому пытается с ними соединиться. Спам-хост будет пытаться отослать как можно больше писем и поэтому устанавливать большое количество соединений в единицу времени с портом 25 (SMTP-порт, на который отправляется почта).
- *Выявление узлов с необычно интенсивным трафиком* (Top N Data). В этом случае хост, который обычно не входил в число *N* самых активных, начинает посылать или получать необычно большое количество данных в единицу времени, то есть генерировать слишком интенсивный трафик. Это также может быть DoS-атака или же активность червя, пытающегося заразить другие хосты.
- *Анализ SYN и других флагов заголовка TCP.* Наличие необычно большого числа пакетов с установленным флагом SYN или другими флагами заголовка TCP может свидетельствовать о DoS-атаке.

- *Анализ ICMP-сообщений.* Большое количество ICMP-сообщений «Порт/хост/сеть недоступен» может свидетельствовать о сканировании злоумышленником или вирусом хостов и портов.

Другим эффективным методом анализа трафика является проверка значений некоторых полей пакетов на предмет совпадения со значениями, используемыми в известных типах атак (*сравнение с образцами*). Чаще всего образцами атаки являются значения *портов TCP/UDP* и *IP-адресов*. Например, червь SQL Slammer чаще всего использует TCP-порт 1434, а червь W32/Netsky.c всегда использует DNS-сервер с адресом из списка конкретных IP-адресов.

Когда коллектор получает каждую секунду данные о тысячах, а иногда и десятках и даже сотнях тысяч потоков, то для обработки таких данных естественно задействовать специальное программное обеспечение. Программные системы анализа данных NetFlow существуют. Они автоматизируют процедуры выявления аномальной активности в сети, проверяя потоки на соответствие многочисленным образцам разнообразных атак, в первую очередь атак отказа в обслуживании и сканирования сети и портов. Данные, отнесенные системой к подозрительной активности, выделяются в особую группу и предоставляются администратору сети в компактной форме. Кроме того, администратор может создавать собственные правила выявления подозрительной активности.

В целом подход к анализу данных NetFlow должен быть адаптивным, основанным на постоянном обновлении и пополнении базы признаков атак, то есть аналитик должен стараться «идти в ногу» с разработчиками вирусов, ботов и другого вредоносного программного обеспечения.

Системы обнаружения вторжений

Система обнаружения вторжений (Intrusion Detection System, **IDS**) — это программное или аппаратное средство, которое выполняет непрерывное наблюдение за сетевым трафиком и деятельностью субъектов системы с целью предупреждения, выявления и протоколирования атак.

> В отличие от файерволов и прокси-серверов, которые строят защиту сети исключительно на основе анализа сетевого трафика, системы обнаружения вторжений учитывают в своей работе различные подозрительные *события*, происходящие в системе.

Существуют ситуации, когда сетевой экран оказывается проницаемым для злоумышленника: например, когда атака идет через туннель VPN из взломанной сети, когда инициатором атаки является пользователь внутренней сети и т. п. И дело здесь не в плохой конфигурации межсетевого экрана, а в самом принципе его работы. Файервол, несмотря на то что обладает памятью и анализирует последовательность событий, конфигурируется на блокирование трафика с *заранее предсказуемыми признаками*, например по IP-адресам или протоколам. Так что факт взлома внешней сети, с которой у него был установлен защищенный канал и которая прежде вела себя вполне корректно, в правилах экрана отразить нельзя. Точно так же, как и неожиданную попытку легального внутреннего пользователя скопировать файл с паролями или повысить уровень своих привилегий. Подобные подозрительные действия может обнаружить только система, оснащенная агентами, встроенными во многие точки сети, причем она должна следить не только за трафиком, но и за всеми обращениями к критически важным ресурсам отдельных компьютеров, а также иметь

информацию о перечне подозрительных действий (сигнатур атак) пользователей. Таковой и является система обнаружения вторжений. Она *не дублирует* действия файервола, а *дополняет* их, производя, кроме того, автоматический анализ всех журналов событий, имеющихся у сетевых устройств и средств защиты, чтобы попытаться найти следы атаки, если ее не удалось зафиксировать в реальном времени.

Другим важным отличием IDS от файерволов является то, что в обязанности IDS *не входит блокировка* подозрительного трафика. IDS только пытается выявить подозрительную активность и поднять тревогу — обычно путем предупреждения администратора сети электронным сообщением. Кроме поднятия тревоги IDS протоколирует подозрительные пакеты, помещая их в журнал.

Типовая система IDS включает следующие функциональные элементы (рис. 28.11):

- источники данных;
- датчики;
- анализатор;
- администратор;
- оператор;
- менеджер.

Рис. 28.11. Элементы функциональной архитектуры IDS

Источниками данных для сетевой системы IDS являются маршрутизаторы, коммутаторы и хосты локальной сети, словом, все элементы сети, которые передают, генерируют и принимают трафик.

Датчик копирует пакеты, циркулирующие в сети, и передает их анализатору для выявления подозрительной активности. Датчик может представлять собой отдельный компьютер, подключенный к зеркализованному порту коммутатора (как это показано на рисунке), или

же это может быть программный компонент маршрутизатора, который имеет доступ к пакетам, буферизуемым на его интерфейсах. Датчик может осуществлять первичную фильтрацию пакетов, отбирая только те пакеты, которые удовлетворяют некоторым очевидным критериям, например направленные к публичным веб-серверам, атакуемым наиболее часто.

Анализатор является «мозгом» IDS, он получает данные от датчиков и проверяет их на наличие угроз и подозрительной активности в сети. Анализатор работает на основе правил, составленных *администратором* системы безопасности предприятия в соответствии с политикой безопасности. При выполнении условия одного из правил анализатор вырабатывает сообщение тревоги и передает его *менеджеру* системы IDS — программному компоненту, который хранит конфигурацию IDS и поддерживает удобный интерфейс с оператором IDS. Менеджер IDS оповещает оператора IDS о тревоге в виде некоторого уведомления, привлекающего внимание, например в виде текстовой строки на экране с мерцающим символом, в виде звукового сигнала, продублированного электронным письмом, и т. п.

Оператор системы IDS на основе данных уведомления принимает решение о реакции сети на подозрительную активность — это может быть отключение сетевого интерфейса, через который поступает подозрительный трафик, изменение правил файервола для блокировки определенных пакетов или же игнорирование уведомления, если оператор считает, что вероятность вторжения очень мала. В любом случае, все данные о потенциальном вторжении протоколируются в журнале менеджера и могут быть использованы впоследствии для повторного анализа ситуации. Если же IDS выполняет также функции IDP, то менеджер может автоматически передать команды на маршрутизатор или файервол для блокировки подозрительного трафика.

Описанная схема является функциональной, в реальной системе IDS эти функции не обязательно реализуются в отдельных блоках или модулях системы. В минимальном варианте все функции IDS могут быть сосредоточены в программном обеспечении единственного компьютера, сетевой адаптер которого исполняет роль датчика за счет того, что присоединен к зеркализованному порту коммутатора или маршрутизатора. Правда, в этом случае контролироваться будет только один сегмент корпоративной сети (или же несколько, если на порт коммутатора, к которому подключен компьютер с IDS, зеркализируется несколько рабочих портов коммутатора, однако портов не может быть много, если скорости всех портов коммутатора равны, так как получающий трафик порт быстро переполнится).

Более масштабируемой является реализация IDS с несколькими датчиками, подключенными к различным сегментам сети и посылающими захваченный трафик центральному анализатору. В качестве таких датчиков может выступать дополнительное программное обеспечение маршрутизатора или коммутатора или же отдельные аппаратные устройства.

Наряду с системами обнаружения вторжений существуют **системы предупреждения вторжений** (Intrusion Prevention Systems, **IDP**), которые выполняют автоматические действия по прекращению атаки в случае ее обнаружения. Часто такие системы перепоручают эту работу файерволу, передавая ему новое правило для блокировки подозрительного трафика.

В IDS для обнаружения вторжений применяются нескольких типов правил:

- *Правила, основанные на сигнатуре (подписи) атаки* (signature rules), используют характерную для атаки последовательность символов в данных пакета. Например, правило может диктовать поиск строки «user root» в полях FTP-пакета — как известно, этот протокол передает пароли пользователей в открытом виде и применение его суперпользователем root считается грубым нарушением политики безопасности предприятия, так

что система IDS должна отслеживать такие случаи. Чаще всего сигнатуры атак относятся к прикладным протоколам, для обнаружения вторжения на транспортном уровне они менее пригодны. Для эффективной работы система IDS должна иметь обширную, постоянно пополняемую базу сигнатур атак.

- *Правила, основанные на анализе протоколов* (protocol rules), связаны с проверкой логики работы протокола и фиксацией отклонений от него. Так как каждый протокол обладает специфической логикой, IDS обычно имеет библиотеку программных модулей, каждый из которых может анализировать поведение определенного протокола. Правила анализа протоколов написать гораздо сложнее, чем правила анализа подписи атаки, так как для этого нужно хорошо знать логику протокола и возможные отклонения от нее. Реализация правил анализа протоколов требует большого быстродействия IDS, в противном случае работа системы IDS может значительно замедлиться и она не сможет работать в реальном времени.
- *Правила, основанные на статистических аномалиях трафика*, проверяют такие характеристики трафика, как Top N sessions, Top N Data, которые были рассмотрены в описании технологии анализа данных NetFlow. В принципе, любая статистика активности пользователей корпоративной сети может служить для этой цели. Например, если 10 % трафика пользователей отдела планирования всегда направлено к серверу базы данных финансового отдела, то появление пользователя, у которого 90 % трафика идет на работу с этим сервером, может вызвать подозрение: возможно, компьютер этого пользователя захвачен злоумышленником, который удаленно пытается похитить важные финансовые данные предприятия.

Архитектура сети с защитой периметра и разделением внутренних зон

Демилитаризованная зона является практически обязательным элементом защищенной архитектуры любой корпоративной сети, однако в общем случае такая сеть должна быть разбита на большее число сегментов со сходными требованиями к защите на основе фильтрации и анализа трафика. Этот подход иллюстрируется архитектурой сети, показанной на рис. 28.12. Здесь достаточно крупная корпоративная сеть разделена на шесть сегментов, каждый из которых представляет отдельную IP-сеть.

Каждый из сегментов сети представляет собой отдельную зону безопасности. Сети зон соединены друг с другом через *корпоративный файервол*, непосредственной связи между этими сетями нет, — такая архитектура позволяет надежно реализовывать правила политики безопасности для каждой зоны. Корпоративный файервол выполнен в виде двух устройств, работающих в режиме горячего резервирования, когда каждое из устройств реализует одни и те же правила фильтрации трафика и в случае отказа одного из устройств работоспособное устройство без разрыва имеющихся соединений может продолжить обслуживать трафик, проходивший ранее через отказавшее устройство.

Корпоративный файервол также контролирует интернет-трафик предприятия, который проходит через две линии связи с различными интернет-провайдерами, — такое соединение достаточно типично для крупных корпоративных сетей, так как оно обеспечивает высокую надежность интернет-связи.

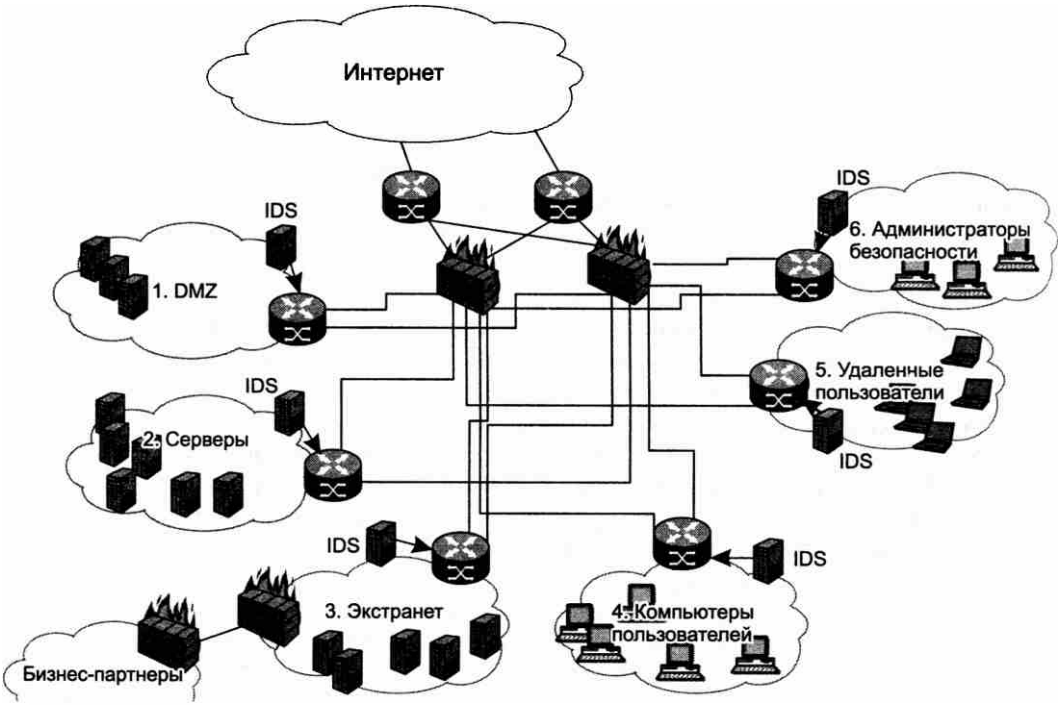


Рис. 28.12. Архитектура сети с несколькими зонами защиты

Посмотрим на состав и требования к защите каждой из зон.

Зона 1 представляет собой демилитаризованную зону предприятия с открытыми для публичного доступа серверами. Ее особенности мы уже рассмотрели. Иногда эту зону разделяют на две зоны, выделяя веб-серверы в отдельную зону, так как веб-сервисы весьма специфичны и их защита на прикладном уровне существенно отличается от защиты почтового сервера или DNS-сервера.

Зона 2 — это зона внутрикорпоративных серверов, иногда называемая корпоративным порталом. Здесь сосредоточены все информационные ресурсы предприятия, к которым обращаются сотрудники в своей работе: внутренний веб-сервер, серверы баз данных, внутренний почтовый сервер, серверы приложений управления предприятием и т. п. К этой зоне должны иметь доступ только пользователи предприятия, доступ внешних пользователей должен быть заблокирован. Но и для внутренних пользователей доступ к ресурсам этой зоны должен быть ограничен файерволом в соответствии с принципом минимальных привилегий. На уровне файервола он означает, что пользователям должен быть разрешен доступ только к портам тех приложений, которые им нужны в работе, а все остальные порты должны быть файерволом заблокированы. Так, если какому-то пользователю нужен доступ только к веб-сервису, то ему должен быть разрешен доступ к определенному серверу зоны 1 по порту 80 и/или 8080, а все остальные порты этого сервера должны быть заблокированы.

Зона 3 — это зона *экстранет* (extranet), где сосредоточены ресурсы, доступ к которым нужен сотрудникам предприятий-партнеров. Эти ресурсы представлены базами данных или веб-приложениями, содержащими конфиденциальные данные, к которым нужно

предоставлять доступ только сотрудникам предприятий-партнеров, а доступ из публичного домена Интернета должен быть запрещен. Зона экстранет в нашем примере соединена с предприятиями-партнерами отдельной линией связи, возможно, через отдельного провайдера, и контролируется отдельным файерволом. В других случаях доступ к экстранет может проходить через общие для всех зон линии связи с Интернетом и контролироваться тем же файерволом. Для обеспечения конфиденциальности данных ресурсов экстранет может быть применена технология *виртуальных частных сетей* (VPN) — в этом случае файервол выполняет также функции VPN-шлюза.

Зона 4 — это внутренняя зона предприятия, в ней находятся клиентские компьютеры сотрудников предприятия. Для этой зоны разрешается установление соединений с серверами зоны 2 (корпоративный портал) и внешними серверами Интернета. Установление соединений с компьютерами этой зоны извне (то есть из Интернета и из любой другой зоны предприятия) запрещается.

Зона 5 объединяет сотрудников предприятия, пользующихся удаленным доступом, то есть работающих из дома или сетей других предприятий или публичных провайдеров Интернета (например, из зон Wi-Fi на вокзалах, аэропортах, кафе и т. п.). Таких пользователей называют мобильными. Обычно для хостов этой зоны устанавливаются те же правила доступа, что и для пользователей зоны 4, то есть они имеют доступ к ресурсам зоны 2 и могут устанавливать соединения с ресурсами Интернета (которые проходят через тот же корпоративный файервол, что и соединения внутренних пользователей). Для обеспечения конфиденциальности, как и в случае с экстранет, доступ мобильных пользователей осуществляется через защищенные каналы VPN.

Зона 6 объединяет серверы и клиентские компьютеры, используемые для администрирования средств безопасности предприятия. Здесь сосредоточены серверы политики файерволов, серверы антивирусной защиты, приложений обеспечения безопасности, таких как анализаторы трафика NetFlow.

К сети каждой зоны подключен сервер системы IDS, который выполняет анализ трафика этой сети (или, по крайней мере, ее наиболее критичных сегментов) и предупреждает оператора систем безопасности о подозрительной активности.

Файерволы корпоративной сети должны быть сконфигурированы так, чтобы их правила отражали политику безопасности предприятия. Собственно, эта политика и должна определять структуризацию ресурсов сети на зоны, приведенный пример — только один из вариантов этой политики, хотя и достаточно типичный. Возможно и другое разбиение ресурсов на зоны — как более детальное, так и более укрупненное. Например, зона 5, объединяющая в нашем примере всех пользователей предприятия, работающих в его локальной сети (то есть не удаленно), может быть достаточно просто разбита на несколько зон, если принять во внимание организационную структуру предприятия. Например, в отдельную зону безопасности может быть выделен финансовый отдел.

После определения количества зон для каждой из них должны быть определены критичные ресурсы, доступ к которым нужно ограничивать, и для каждого ресурса (а это, как правило, серверная часть некоторого сетевого сервиса) должны быть определены группы пользователей, которым обеспечивается доступ к ресурсу, остальным пользователям доступ должен блокироваться.

Способ определения групп пользователей зависит от типа файервола и его функциональных возможностей. Для файерволов на основе маршрутизаторов эти группы должны за-

даваться диапазонами IP-адресов, так как другие типы объектов, как правило, списками доступа маршрутизаторов не поддерживаются. Поэтому здесь, по сути, пользователи отождествляются с компьютерами, на которых они работают, более дифференцированной и надежной идентификации этот тип файервола не обеспечивает.

В то же время файерволы высокого уровня, основанные на специальном программном и аппаратном обеспечении, могут использовать те же идентификаторы пользователей и групп пользователей, что и системы аутентификации и авторизации операционных систем или централизованных справочных служб типа Microsoft Active Directory. В этом случае правила файервола оказываются более избирательными и эффективными, так как они оперируют реальными пользователями, прошедшими аутентификацию.

Аудит событий безопасности

Аудит (auditing) — это набор процедур учета и анализа всех событий, представляющих потенциальную угрозу для безопасности системы.

Аудит является одной из задач, решаемых в целях обеспечения важнейшего требования безопасности — подотчетности. В государственном стандарте **подотчетность** определяется как свойство безопасной системы, «обеспечивающее однозначное прослеживание действий любого логического объекта». Возможность фиксировать деятельность объектов системы, а затем ассоциировать их с индивидуальными идентификаторами пользователей позволяет выявлять нарушения безопасности и определять ответственных за эти нарушения.

Для обеспечения свойства подотчетности в компьютерных сетях используются различные программно-аппаратные средства, способные анализировать состояние и параметры элементов системы. К таким средствам относятся уже рассмотренные нами файерволы, системы обнаружения и предотвращения вторжений, анализаторы протоколов, антивирусные системы, а также некоторые подсистемы ОС.

Аудит позволяет «шпионить» за выбранными объектами и выдавать сообщения тревоги, когда, например, какой-либо рядовой пользователь пытается прочитать или модифицировать системный файл. Если кто-то пытается выполнить действия, отслеживаемые системой безопасности, то система аудита пишет сообщение в журнал регистрации, идентифицируя пользователя.

Журнал регистрации — это совокупность хронологически упорядоченных записей о специально отобранных событиях. Данные журнала регистрации должны быть надежно защищены от модификации и разрушения неавторизованными субъектами, чтобы не дать возможности нарушителям «стереть следы» их преступной деятельности.

Как мы видим, аудит по своей природе является *реактивным* (а не проактивным) действием, то есть записи становятся достоянием специалиста по безопасности уже по прошествии некоторого времени после того, как события произошли.

Поскольку никакая система безопасности не гарантирует защиту на уровне 100 %, то именно система аудита оказывается последним рубежом в борьбе с нарушениями.

Эту мысль изящно выражает популярное изречение «Prevention is ideal, but detection is a must!», которое говорит о том, что, бесспорно, предупреждение нарушений — это жела-

тельная, но, увы, часто недостижимая цель, в то время как обнаружение и фиксация нарушений — это то, что должно быть сделано обязательно.

Действительно, после того как злоумышленнику удалось провести успешную атаку, пострадавшей стороне не остается ничего другого, как обратиться к службе аудита. Если при настройке службы аудита были правильно заданы события, которые требуется отслеживать, то подробный анализ записей в журнале может дать много полезной информации. Эта информация, возможно, позволит найти злоумышленника или, по крайней мере, предотвратить повторение подобных атак устранением уязвимых мест в системе защиты. Аудит действует также как *угроза* потенциальным нарушителям, предупреждая их о том, что в случае несанкционированных действий они легко могут быть выведены на «чистую воду».

Данные аудита могут использоваться для разрешения правовых споров: например, сотрудники предприятия могут посчитать некоторые действия по сбору данных нарушающими их законные права. Закон может не разрешить администрации сети фиксацию реального имени пользователя при трассировке его обращений к тем или иным сайтам Интернета. Но даже если аудит не выходит за рамки закона, администрация предприятия должна учитывать тот факт, что в результате неправильно реализованной политики аудита на предприятии может сложиться атмосфера «слежки», которая отрицательно сказывается на эффективности работы предприятия. Во избежание этого рекомендуется заранее предупреждать пользователей сети о том, какие из их действий регистрируются.

Стартовой точкой процесса накопления данных может быть как осуществление того или иного события, так и наступление некоего заранее заданного момента, когда от системы требуется выполнение определенных действий по сбору данных. В некоторых случаях это «расписание» является настолько плотным, что процесс сбора данных о состоянии системы можно назвать *непрерывным*. Функциональный компонент аудита, обеспечивающий непрерывное наблюдение за параметрами системы, называется подсистемой **мониторинга**.

Мы уже обсуждали мониторинг сетевого трафика, кроме него объектами мониторинга могут выступать также загрузка процессора, статус сетевого интерфейса (активный или пассивный), включенное или выключенное устройство печати. Специфическим видом мониторинга является мониторинг нажатия клавиш пользователем. Это средство должно применяться с большой осторожностью и только при проведении расследований, когда имеется явный подозреваемый.

Ведение журнала событий, особенно в режиме мониторинга, может потребовать слишком много ресурсов вычислительной системы (дискового пространства, вычислительной мощности процессора), а также рабочего времени персонала. Поэтому очень важно соблюдать баланс между количеством различных видов событий, подлежащих регистрации, с одной стороны, и затрачиваемыми на их протоколирование ресурсами и возможностью их анализа — с другой.

Отбор событий должен учитывать предысторию процесса функционирования системы и специфику текущей ситуации. Задачу формирования правил отбора событий, подлежащих регистрации, выполняет администратор сети. В ОС, например, к числу таких событий относят попытки успешного и неуспешного логического входа в систему, запуска программ, доступа к защищаемым ресурсам, изменения атрибутов объектов и полномочий пользователей и др. Практически всегда фиксируются события, важные для операционной системы в целом: рестарт системы, очистка журнала регистрации событий, изменение системного времени и многие другие. Любое из этих событий может свидетельствовать об

атаке: например, очистка журнала безопасности может говорить о том, что злоумышленник пытался уничтожить следы атаки. Аудиту должны подлежать не только события, инициированные пользователями, но и действия администраторов сети, которые тоже должны быть подотчетны наравне со всеми. Поэтому время от времени аудит должен проводиться сторонними организациями.

ПРИМЕЧАНИЕ

Термин «аудит безопасности» может использоваться в более широком смысле для обозначения процедур анализа событий, угрожающих безопасности не только компьютерной сети, но и информационной системы предприятия в целом. В этом случае аудит включает обследование IT-инфраструктуры, оценку принятых на предприятии политик безопасности и реализующих эти политики инструкций. Важную часть аудита составляет оценка рисков.

Анализ собранных данных может оказаться трудной задачей уже при продолжительности периода наблюдений в несколько дней. Даже самые простые журналы событий содержат такое огромное количество сведений, что их практически невозможно анализировать «вручную», без специальных средств. Для автоматизации обработки и анализа данных журнала регистрации могут быть использованы следующие средства:

- *средства предварительной обработки данных аудита* предназначены для сжатия информации журнала регистрации за счет удаления из него малоинформативных записей, которые только создают ненужный «шум»;
- *средства выявления аномальных ситуаций* заключаются в постоянном отслеживании среднестатистических *характеристик* системы с последующим их сравнением с текущими значениями: например, факт входа в систему в часы, не типичные для режима работы какого-либо сотрудника, уже является поводом для выдачи предупреждения администратору;
- *средства распознавания атак по их сигнатурам*, то есть по характерным признакам атак, выражающимся в виде специфического фрагмента кода, типичного поведения, аномально частого обращения к какому-либо сетевому порту компьютера.

При реализации аудита в больших сложных системах, состоящих из множества подсистем с собственными средствами протоколирования событий, иногда необходимо приложить специальные усилия по «синхронизации» журналов регистрации. В частности, поскольку в разных частях большой системы одни и те же объекты могут иметь разные имена и, напротив, разные объекты — одинаковые имена, администраторам, возможно, придется принять специальное соглашение об однозначном именовании объектов.

Выводы

Под фильтрацией понимается обработка IP-пакетов маршрутизаторами и файерволами, приводящая к отбрасыванию некоторых пакетов или изменению их маршрута. Выборочная передача кадров/пакетов маршрутизатором осуществляется на основе стандартных и дополнительных правил, называемых также фильтрами.

Файервол — это программно-аппаратный комплекс, выполняющий разнообразные функции по защите внутренней сети, набор которых может меняться в зависимости от типа,

модели и конкретной конфигурации файервола, при этом минимальный набор функций должен включать фильтрацию трафика для предотвращения сетевых атак и аудит событий, связанных с фильтрацией.

Прокси-сервер — это особый тип приложения, которое выполняет функции посредника между клиентскими и серверными частями распределенных сетевых приложений, причем предполагается, что клиенты принадлежат к внутренней (защищаемой) сети, а серверы — к внешней (потенциально опасной) сети.

Трансляция сетевых адресов (NAT) — это особый вид обработки трафика, при котором внешний IP-адрес пакета, который использовался при маршрутизации пакета через Интернет, заменяется внутренним, предназначенным для маршрутизации во внутренней сети.

Применение NAT позволяет преодолеть дефицит адресов IPv4, а также скрыть адреса узлов внутренней сети, чтобы не дать злоумышленникам возможность составить представление о ее структуре.

Для надежной и эффективной защиты корпоративной сети она должна быть логически сегментирована таким образом, чтобы ресурсы каждой подсети в отношении мер защиты были подобными. Ресурсы корпоративной сети, к которым обращаются внешние пользователи, размещаются в демилитаризованной зоне.

Мониторинг сетевого трафика — это непрерывный процесс инструментального автоматизированного наблюдения за отдельными параметрами трафика с целью проверки соблюдения SLA, планирования сети, а также предотвращения негативных событий, таких как технические аварии, угрозы и атаки злоумышленников. Мониторинг трафика осуществляют анализаторы протоколов, маршрутизаторы, поддерживающие протокол NetFlow, системы обнаружения вторжений.

В отличие от файерволов и прокси-серверов, которые строят защиту сети исключительно на основе анализа сетевого трафика, системы обнаружения вторжений (IDS) учитывают в своей работе различные подозрительные события, происходящие в системе.

В IDS для обнаружения вторжений применяются нескольких типов правил:

- правила, основанные на сигнатуре (подписи) атаки;
- правила, основанные на проверке логики работы протокола и фиксации отклонений от него;
- правила, основанные на статистических аномалиях трафика.

Аудит — это набор процедур учета и анализа всех событий, представляющих потенциальную угрозу для безопасности системы. Аудит не позволяет предотвратить нарушение, но позволяет обнаружить и зафиксировать его.

Контрольные вопросы

1. Основными функциями файервола являются:
   а) аутентификация;
   б) авторизация;
   в) аудит;
   г) фильтрация трафика.

2. Какие из следующих утверждений о технологии NAT справедливы? Варианты ответов:
   а) устройство, поддерживающее NAT, заменяет внешние IP-адреса пакетов, которые использовались при маршрутизации пакета через Интернет, внутренними;
   б) устройство, поддерживающее NAT, фильтрует входящий трафик, отбрасывая все пакеты с частными IP-адресами назначения;
   в) причина обращения к технологии NAT — дефицит адресов IPv4;
   г) причина обращения к технологии NAT — скрытие адресов хостов для повышения безопасности сети.
3. Какова должна быть защита общедоступного сервера (компьютера-бастиона)? Варианты ответов:
   а) поскольку доступ к этому компьютеру открыт, его помещают в демилитаризованную зону, поэтому он не требует защиты;
   б) уровень защиты компьютера-бастиона такой же, как у серверов внутренней сети;
   в) компьютер-бастион должен быть особенно тщательно защищен от внешних пользователей;
   г) правила, определенные для компьютера-бастиона по доступу к ресурсам внутренней сети, должны быть более строгими, чем правила, регламентирующие доступ к нему внешних пользователей.
4. В чем состоят главные отличия системы обнаружения вторжений (IDS) от файерволов с запоминанием состояния сеанса? Варианты ответов:
   а) IDS не блокирует подозрительный трафик;
   б) IDS анализирует не только события, связанные с прохождением трафика, но и другие подозрительные события в сети;
   в) IDS не выполняет журнализацию событий;
   г) в отличие от файерволов системы IDS всегда являются исключительно программными.
5. Справедливо ли следующее утверждение: «Прокси-сервер всегда работает в режиме запоминания состояния сеанса»?
6. В этой главе рассмотрен пример архитектуры сети с защитой периметра и разделением внутренних зон. Предложите альтернативный вариант.

ГЛАВА 29 Атаки на транспортную инфраструктуру сети

TCP-атаки

Протокол TCP используется злоумышленниками и как инструмент для организации атак (обычно это атаки отказа в обслуживании), и как цель нападения — нарушение TCP-сеанса атакуемого приложения, например путем подделки сегмента.

Затопление SYN-пакетами

Этот тип DoS-атаки активно применяется злоумышленниками на протяжении многих лет; впервые он был подробно описан (с приведением кода атаки) в 1996 году, и уже в том же году началось его практическое «применение», которое продолжается по сей день. Атакуемым является конечный узел, как правило, сервер, работающий с клиентами по протоколу TCP.

Атака **затоплением SYN-пакетами** (SYN Flood) использует уязвимость процедуры установления логического соединения протокола TCP. Как мы уже обсуждали в главе 16, эта процедура основана на использовании флагов SYN и ACK, переносимых в заголовке каждого TCP-сегмента (рис. 29.1, *а*). Для реализации атаки злоумышленник организует передачу на сервер массированного потока пакетов с флагом SYN, каждый из которых инициирует создание нового TCP-соединения (рис. 29.1, *б*). Получив пакет с флагом SYN, сервер выделяет для нового соединения необходимые ресурсы и в полном соответствии с протоколом отвечает клиенту пакетом с флагами *ACK* и *SYN*. После этого, установив тайм-аут, он начинает ждать от клиента завершающий пакет с флагом *ACK*, который, увы, так и не приходит.

Аналогичным образом создается множество других «недоустановленных» соединений. Обычно операционная система сервера имеет лимит на количество одновременно поддерживаемых «недоустановленных» TCP-соединений (глобально или для каждого программного порта отдельно), так как каждое открытое соединение требует выделения памяти ядра ОС для нового **блока TCB** (Transmit Control Block). Этот блок содержит данные о состоянии соединения: сокет клиента, номер ожидаемого сегмента, указатель на положение сегмента в буфере и др. Блок TCB имеет размер от 280 до 1300 байт в зависимости от типа ОС. При достижении лимита ОС начинает отвергать все последующие запросы на установление TCP-соединений, а следовательно, отказывает в обслуживании всем, в том числе легальным клиентам сервера. По истечении тайм-аута ОС удаляет из памяти блоки TCB «недоустановленных» соединений и снова начинает устанавливать новые соединения.

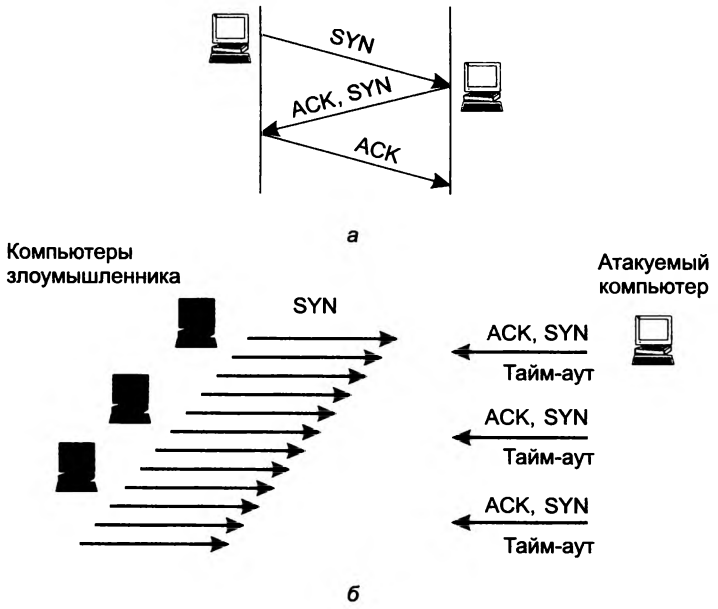


Рис. 29.1. Проведение DoS-атаки, в которой используются особенности протокола TCP: *а* — нормальный порядок установления TCP-соединения; *б* — DoS-атака путем создания множества незакрытых TCP-соединений

Для осуществления атаки затоплением SYN-пакетами атакующий должен заблокировать нормальную реакцию своего компьютера на получение от атакуемого сервера сегмента с флагами SYN/ACK. Нормальная реакция состоит в том, что в соответствии с протоколом TCP атакующий должен отправить в ответ сегмент с флагом ACK. Но если это произойдет, атакуемый сервер посчитает процедуру установления TCP-соединения завершенной, удалит соответствующий блок TCB из списка «недоустановленных» соединений и начнет принимать новые соединения, то есть атака не удастся. Поэтому атакующий фильтрует входящий трафик, отсеивая ответы SYN/ACK от атакуемого сервера.

Обычно атака затоплением SYN-пакетами обнаруживается за счет наличия в трафике большого количества SYN-сегментов без соответствующего количества ACK-сегментов, идущих от того же источника. При этом заметного всплеска сетевого трафика может и не быть, так как лимит «недоустановленных» соединений сам по себе не столь велик. Главным средством борьбы с атакой затоплением SYN-пакетами является *фильтрация* трафика, поступающего от источника атаки. Для этого нужно определить адрес атакующего узла.

В некоторых случаях это сделать не просто, так как атакующий может помещать SYN-сегменты в IP-пакеты с «поддельным» адресом отправителя. В этом случае говорят, что он использует *спуфинг*. Спуфинг помогает атакующему не только преодолеть защитный фильтр, но и избавиться от вредящих ему ответных SYN/ACK-сегментов атакуемого сервера. Для этого ему достаточно выбрать в качестве «поддельных» такие адреса, которые не будут реагировать на SYN/ACK-сегменты, например адреса несуществующих узлов.

ПРИМЕЧАНИЕ

Спуфинг IP-адресов источника используется во многих типах атак, поэтому борьба с ним — естественный элемент обеспечения сетевой безопасности. Основным средством борьбы является применение на маршрутизаторах техники проверки обратного пути (Reverse Path Check, RPC). Идея этой проверки достаточно проста: пакет должен передаваться маршрутизатором в соответствии с его адресом назначения только в том случае, если его адрес источника имеется в таблице маршрутизации для интерфейса, с которого этот пакет получен. Действительно, если компьютер злоумышленника подключен к сети 212.100.100.0/24, но генерирует пакеты с адресом источника 25.0.30.18, то маршрутизатор провайдера, к которому подключена сеть 212.100.100.0/24, легко может проверить, что через интерфейс, на который был получен пакет с подделанным адресом, достичь сети 25.0.30.18 нельзя, а значит, пакет нужно отбросить. Однако техника RPC работает не всегда. В тех случаях, когда сеть злоумышленника имеет несколько подключений к сетям разных провайдеров, она может приводить к отбрасыванию легитимных пакетов.

Преодоление атаки путем фильтрации также осложняется, когда поток SYN-сегментов поступает на атакуемый сервер сразу от сотен зараженных компьютеров какой-нибудь сети ботов, то есть когда имеет место **распределенная атака затоплением SYN-пакетами** (DDoS SYN Flood).

Другим способом борьбы с атакой затоплением SYN-пакетами является *изменение параметров протокола TCP* — увеличение предельного числа «недоустановленных» соединений, уменьшение тайм-аута вытеснения старых «недоустановленных» соединений, усложнение логики самой процедуры установления соединения, например введение специальных *cookie-блоков SYN*. В этом методе при приеме запроса SYN-сервер не запоминает блок TCB в своей оперативной памяти, а посылает его (в сжатом виде) клиенту вместе с SYN/ACK-ответом. При нормальном ходе установления соединения клиент отвечает ACK-сегментом, в котором повторяет сжатый блок TCB, сервер, получив этот ACK-сегмент, а с ним и все параметры устанавливаемого соединения, создает соответствующий блок TCB в памяти своего ядра. Поскольку в этой модифицированной процедуре на начальном этапе установления соединения ресурсы на сервере не выделяются, то и атака затоплением SYN-пакетами просто не получается.

Разновидностью TCP-атаки затоплением SYN-пакетами является TCP-атака затоплением ACK-пакетами, выполняемая путем **отражения**. Злоумышленник посылает SYN-пакеты с адресом жертвы на большое количество серверов, которые отвечают на SYN-пакеты пакетами с установленным битом *ACK*. ACK-пакеты бомбардируют атакуемый компьютер и исчерпывают пропускную способность его входного интерфейса. Этот прием превращает DoS-атаку в DDoS-атаку без использования сети ботов, так как все компьютеры, отвечающие на SYN-запросы, не заражаются предварительно каким бы то ни было вирусом, а работают в полном соответствии со стандартной версией протокола TCP.

Подделка TCP-сегмента

Протокол TCP служит для повышения надежности транспорта, для этого каждый сегмент данных сопровождается порядковым номером первого байта сегмента, причем начальные значения этих номеров для каждой из двух сторон, обменивающихся данными, выбираются случайным образом. При приеме очередного сегмента протокол TCP проверяет, находится ли его порядковый номер в пределах окна приема, и только в случае положительного результата такой проверки добавляет принятые данные к байтам, принятым ранее в ходе

данного TCP-сеанса. Описанная проверка предназначена для защиты сегментов некоторого TCP-сеанса от смешивания с сегментами других сеансов, но этот механизм защиты не так уж надежен, чем и пользуются злоумышленники.

Атака **подделкой TCP-сегмента** состоит в генерации TCP-сегментов, все атрибуты которых имеют значения, легитимные для некоторого существующего TCP-сеанса атакуемого компьютера, то есть IP-адреса, номера TCP-портов источника и приемника, а также порядковые номера из текущего диапазона окна приема. Принимающая сторона не может отличить такие поддельные сегменты от настоящих и помещает информацию злоумышленника в поток пользовательских данных, а значит, злоумышленник может добиться желаемого эффекта: например, поместить ложную информацию в базу данных, заразить атакуемый компьютер вирусом и т. п.

Для того чтобы «поддельный» сегмент выглядел как настоящий, атакующий может либо прослушивать трафик, либо просто перебирать все возможные значения адресов, портов и порядковых номеров сегментов. Прослушивание трафика представляет собой самостоятельную нетривиальную задачу, связанную с перенаправлением трафика; атаки такого типа мы рассмотрим позже. В то же время перебор параметров TCP-сеанса требует большой вычислительной мощности компьютера атакующего, но это в последнее время не является проблемой. В обоих случаях атаковать проще длительные TCP-сеансы, например сеансы загрузки больших видеофайлов; короткие сеансы веб-серфинга намного менее уязвимы.

Разновидностью подделки TCP-сегментов является их *повторное использование*. Если злоумышленник смог каким-то образом перехватить трафик между двумя участниками TCP-сеанса, то впоследствии он может просто посылать участникам сеанса дубликаты перехваченных сегментов. Этот прием может применяться злоумышленником для разных целей: например, он может вызвать таким образом нарушение работы некоторого приложения, пользующегося TCP как транспортом, за счет представления устаревшей (перехваченной) информации в качестве новой.

Сброс TCP-соединения

Атака **сбросом TCP-соединения** используется для разрыва TCP-соединений легальных пользователей. При поступлении TCP-сегмента с установленным флагом *RST* узел должен немедленно завершить сеанс, к которому относится этот сегмент, и удалить все данные, полученные в ходе сеанса. Разработчики протокола TCP ввели этот флаг для обработки аварийных ситуаций. Например, если в одном из узлов во время TCP-сеанса происходит сбой, то после восстановления системы он может послать сегмент с этим признаком, чтобы уведомить узел-собеседник о невозможности продолжения сеанса.

Для проведения атаки злоумышленник должен подделать заголовок TCP-сегмента.

Интересно, что прием сброса соединения используется не только злоумышленниками, но и разработчиками средств защиты: например, некоторые файерволы применяют его для прекращения атаки. Известен также случай, когда провайдер (Comcast) с его помощью пытался бороться с программами обмена файлами, работающими на пользовательских компьютерах (и нарушающими авторские права владельцев аудио- и видеозаписей). Правда, через некоторое время Федеральная комиссия США по связи рассмотрела эту практику и признала ее незаконной.

Борьба с атаками подделкой TCP-сегмента и сбросом TCP-соединения может вестись по двум направлениям. Первое направление связано с предотвращением прослушивания трафика, второе основано на изменении поведения самого протокола TCP, например путем включения дополнительной процедуры аутентификации каждого TCP-сегмента с использованием цифровой подписи. Как мы знаем, цифровая подпись не обеспечивает конфиденциальности, так как содержимое защищаемых полей не шифруется, но она гарантирует, что TCP-сегмент не был изменен третьей стороной.

ICMP-атаки

Перенаправление трафика

Перенаправление трафика можно осуществить в самых разных целях, с одной из них, например, мы только что столкнулись, рассматривая атаку подделкой TCP-сегментов.

Способов перенаправления трафика существует несколько. Так, в пределах локальной сети эту задачу можно решить с помощью протокола ICMP. В соответствии с данным протоколом маршрутизатор посылает хосту непосредственно присоединенной локальной сети ICMP-сообщение о перенаправлении маршрута при отказе этого маршрута или в тех случаях, когда обнаруживает, что для некоторого адреса назначения хост использует нерациональный маршрут. На рис. 29.2 применяемый по умолчанию маршрутизатор R1, получив от хоста H1 пакет, адресованный хосту H2, определяет, что наилучший маршрут к хосту H2 пролегает через другой маршрутизатор данной локальной сети, а именно через маршрутизатор R2. Маршрутизатор R1 отбрасывает полученный пакет и помещает его заголовок в ICMP-сообщение о перенаправлении маршрута, которое посылает хосту H1. В сообщении содержится IP-адрес альтернативного маршрутизатора R2, который теперь должен использовать хост, посылая данные хосту H2. Хост H1 вносит изменения в свою таблицу маршрутизации и с этого момента отправляет пакеты хосту H2 по новому, скорректированному маршруту.

Для перехвата трафика, направляемого хостом H1 хосту H2, злоумышленник должен сформировать и послать хосту H1 пакет, маскирующийся под ICMP-сообщение о перенаправлении маршрута (рис. 29.3). В этом сообщении содержится запрос о корректировке таблицы маршрутизации хоста H1 так, чтобы во всех пакетах с адресом IP_{H2} адресом следующего маршрутизатора стал адрес IP_{HA}, являющийся адресом хоста-злоумышленника HA.

Для того чтобы хост «поверил» этому сообщению, в поле IP-адреса отправителя должен быть помещен адрес предлагаемого по умолчанию маршрутизатора R1. Когда пакеты, передаваемые введенным в заблуждение хостом, начнут поступать на узел злоумышленника, он может либо захватывать и не передавать эти пакеты дальше, имитируя для поддержания диалога приложение, которому эти пакеты предназначались, либо организовать транзитную передачу данных по указанному адресу назначения IP_{H2}. Читая весь трафик между узлами H1 и H2, злоумышленник получает все необходимую информацию для несанкционированного доступа к серверу H2.

Сами маршрутизаторы также могут реагировать на ICMP-сообщения о перенаправлении маршрута, но обычно провайдеры отключают эту опцию для предотвращения атак данного типа.

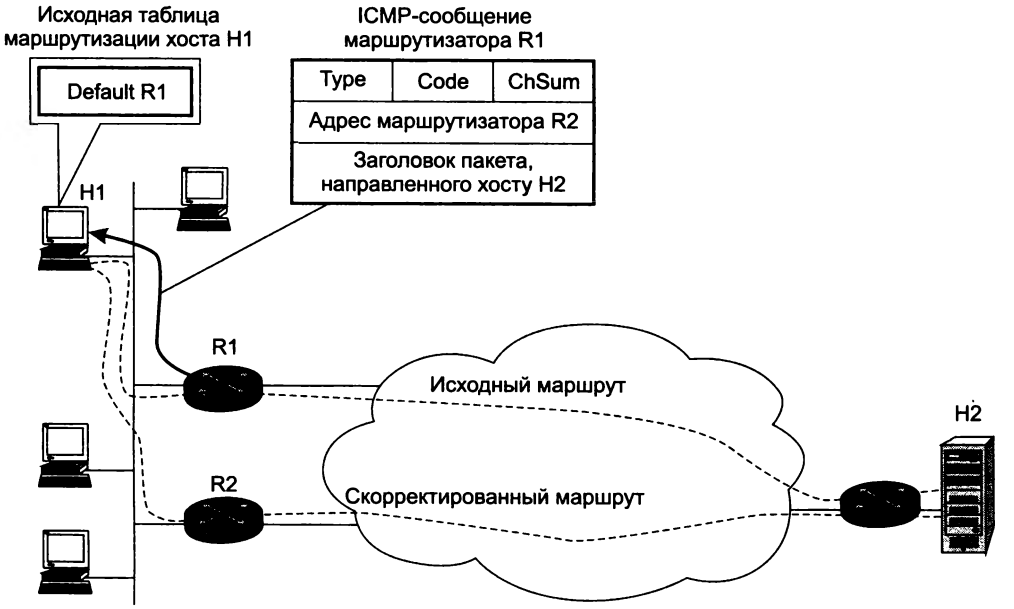


Рис. 29.2. Перенаправление маршрута предлагаемым по умолчанию маршрутизатором

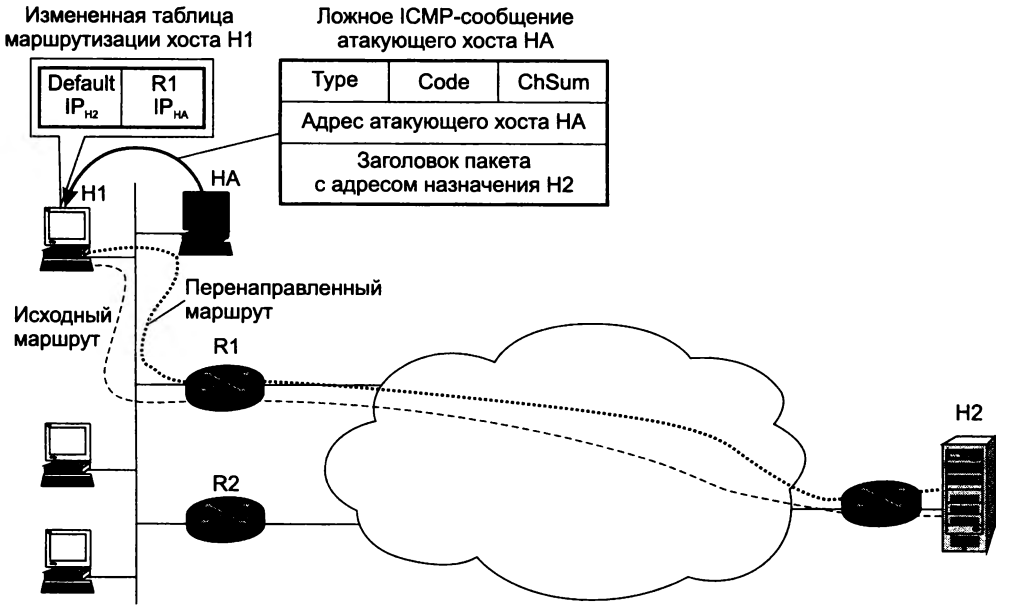


Рис. 29.3. Перенаправление маршрута злоумышленником

Заметим, что простейший вариант перенаправления трафика в локальной сети может быть осуществлен путем отправки в сеть *ложного ARP-ответа*. В данном случае схема очевидна: получив широковещательный ARP-запрос относительно некоторого IP-адреса, злоумышленник посылает ложный ARP-ответ, в котором сообщается, что данному IP-адресу соответствует его собственный MAC-адрес.

ICMP-атака Smurf

ICMP-атака **Smurf** — это DDoS-атака, использующая функцию *эхо-запроса* протокола ICMP. Название атаки произошло от имена файла smurf.c, содержащего код атаки и получившего распространение в 1998 году.

Эхо-запросы и эхо-ответы протокола ICMP больше известны по утилите *ping*[1], с помощью которой можно проверить достижимость узла Интернета. Для проверки достижимости утилита ping посылает тестируемому узлу ICMP-пакет, в котором в качестве типа сообщения указан код 8 (эхо-запрос). Получив его, тестируемый узел отправляет в обратном направлении ICMP-пакет с кодом 0 (эхо-ответ).

В атаке Smurf используется тот факт, что эхо-запрос может быть послан не только по индивидуальному, но и по *широковещательному* (broadcast) адресу некоторой сети. Например, если у сети адрес 200.200.100.0/24, то ее широковещательным адресом является 200.200.100.255, и эхо-запрос должен быть доставлен всем узлам этой сети.

Атаку иллюстрирует рис. 29.4.

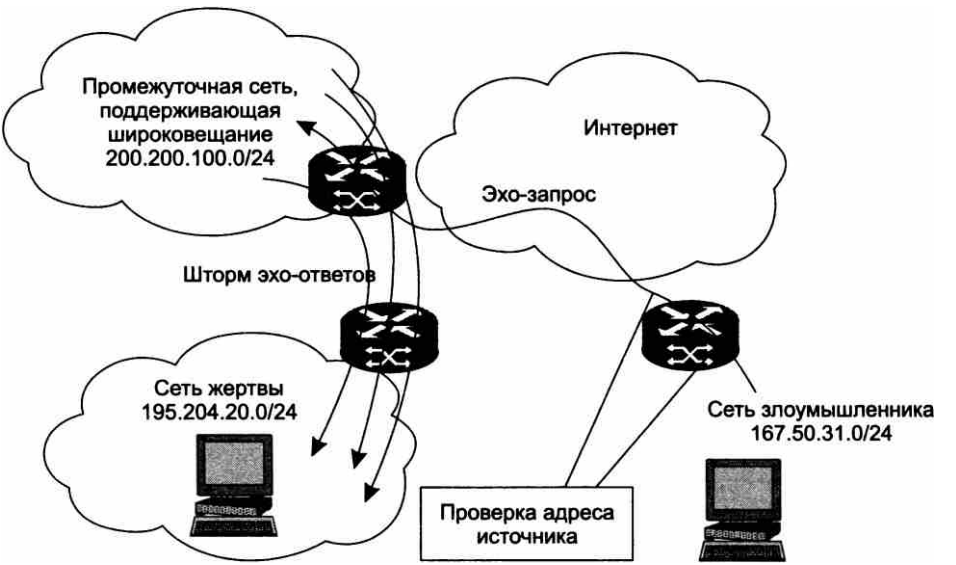


Рис. 29.4. Компоненты ICMP-атаки Smurf

[1] См. раздел «Утилита ping» в главе 15.

Компьютер злоумышленника с адресом 167.50.31.17 находится в сети 167.50.31.0/24, а атакуемый компьютер имеет адрес 195.204.20.145 и подключен к сети 195.204.20.0/24. Компьютер злоумышленника генерирует эхо-запросы с адресом приемника 200.200.100.255 и адресом источника 195.204.20.145. Эхо-запросы передаются через Интернет в сеть 200.200.100.0.24 и принимаются всеми узлами этой сети, которые отвечают на ICMP-запросы эхо-ответами. В том случае, когда в сети 200.200.100.255 имеется достаточно большое количество активных узлов (понятно, что их не может быть более 254), то на атакуемый узел 195.204.20.145 приходит интенсивный поток эхо-ответов, так как именно его адрес указан в эхо-запросах как адрес источника. В результате сетевой интерфейс атакуемого компьютера оказывается затопленным эхо-ответами и при превышении интенсивности этого потока некоторой величины его пропускная способность оказывается исчерпанной.

В ICMP-атаке Smurf используется характерный прием — *усиление атаки за счет отражения* посланного пакета большим количеством компьютеров.

ICMP-атака Smurf представляет сегодня скорее исторический интерес. До 1999 года передача через Интернет IP-пакета с широковещательным адресом была обязательной для маршрутизаторов Интернета, но из-за атак, подобных Smurf, в стандарты было внесено изменение, и сегодня предлагаемым по умолчанию режимом является *фильтрация пакетов с широковещательными адресами*. Кроме того, промежуточная сеть, узлы которой используются для отражения эхо-запроса, может быть экранирована с помощью файервола от эхо-запросов, приходящих из внешних сетей. А чтобы предотвратить ICMP-атаку Smurf из внутренней сети, можно запретить всем компьютерам этой сети реагировать на широковещательные эхо-запросы.

Пинг смерти и Ping-затопление

Атака с несколько драматическим названием **Пинг смерти** (Ping of Death) состоит в отправке на атакуемый компьютер эхо-запроса с длиной IP-пакета, превышающей его максимально возможный размер, который согласно стандарту составляет 65 535 байт. Поскольку соответствующий буфер ядра ОС не рассчитан на такой размер, ОС терпит крах, отсюда и такое название атаки. Так как данная атака основана на превышении размера буфера при сборке фрагментированного IP-пакета, то она является частным случаем атак, использующих IP-фрагментацию (см. далее).

Атака Пинг смерти уже давно имеет только исторический интерес, так как разработчики ОС в середине 90-х годов ввели в стек IP необходимую проверку длины собираемого фрагментированного IP-пакета и тем самым ликвидировали саму базу для атаки.

Другая атака, называемая **Ping-затоплением**, также является достаточно простой: злоумышленник использует утилиту ping своей операционной системы для отправки эхо-запросов на атакуемый компьютер с максимально возможной частотой. Если быстродействие сетевого интерфейса его компьютера выше, чем у атакуемого компьютера, то атака удается, так как вся входная пропускная способность интерфейса атакуемого компьютера оказывается исчерпанной. К тому же атакуемый компьютер будет успевать отвечать на часть эхо-запросов эхо-ответами, что приведет к частичному исчерпанию пропускной способности в выходном направлении, а также к замедлению работы программ из-за отвлечения центрального процессора на обработку эхо-запросов.

UDP-атаки

UDP-затопление

UDP-затопление относится к DoS-атакам и имеет целью исчерпание пропускной способности интерфейса атакуемого компьютера. Она подобна только что рассмотренной атаке Ping-затопления, когда злоумышленник просто направляет интенсивный поток UDP-дейтаграмм на атакуемый компьютер. Поскольку протокол UDP работает без установления соединения, то атакуемый компьютер обязан принимать все направляемые ему UDP-дейтаграммы — он не может, как это делается при обмене данными по протоколу TCP, заставить передающий компьютер ограничить интенсивность потока направляемых ему пакетов, уменьшив размер окна приема. Злоумышленник может использовать аппаратный генератор трафика для того, чтобы генерировать UDP-трафик с максимально возможной скоростью выходного интерфейса, игнорируя ответные ICMP-сообщения в тех случаях, когда у атакуемого компьютера программный порт, указанный в UDP-пакетах, не открыт.

ПРИМЕЧАНИЕ

Однажды один из авторов этого курса нечаянно «организовал» подобную атаку, участвуя в тестах по исследованию качества обслуживания разных классов трафика. В одном из тестов UDP-трафик интенсивности около 90 Мбит/с вместо хоста специально сконфигурированной для теста подсети был по ошибке направлен на хост библиотеки Университета в Саусгемптоне. Так как локальная сеть библиотеки была подключена к интерфейсу маршрутизатора со скоростью 100 Мбит/с (это было довольно давно, в 2004 году, сегодня такое подключение библиотеки не сможет удовлетворить пользователей и без DoS-атаки), то длившийся 15 минут UDP-поток вызвал настолько заметное замедление работы браузеров посетителей библиотеки, что некоторые особенно требовательные из них тут же позвонили своему сетевому администратору, который немедленно зафиксировал этот инцидент как реальную атаку.

Слабым местом такого вида атак является то, что их интенсивность принципиально ограничена производительностью интерфейса атакующего компьютера. Имея стандартный для пользовательского компьютера интерфейс 1 Гбит/с, невозможно затопить UDP-пакетами сервер с интерфейсом 10 Гбит/с. Злоумышленник может преодолеть это ограничение, если у него в распоряжении имеется сеть ботов. Именно такой подход был использован в 2007 году, когда была осуществлена массированная DDoS-атака UDP-затоплением на корневые DNS-серверы, при этом трафик создавался примерно пятью тысячами ботов. Эта атака рассмотрена далее в разделе «Атаки на DNS».

ICMP/UDP-затопление

Атака **ICMP/UDP-затоплением** имеет двойное имя, так как в ней используется два протокола. Злоумышленник направляет интенсивный поток UDP-пакетов, в которых в качестве адреса источника указан адрес компьютера-жертвы, на программные порты компьютеров, находящиеся в пассивном состоянии (то есть в данный момент с этими портами не связаны приложения, слушающие сеть). При получении UDP-пакета с номером пассивного порта компьютер в соответствии с логикой работы стека TCP/IP отвечает ICMP-сообщением о недостижимости порта назначения, которое направляется атакуемому компьютеру. Как

видно из описания, в атаке имеет место отражение от компьютеров промежуточной сети; в случае использования широковещательного адреса она становится DDoS-атакой. Для предотвращения этой атаки применяют те же меры, что и для предотвращения ICMP-атаки Smurf, дополнительно реализуется пропуск файерволом только тех UDP-пакетов, порты которых соответствуют активным приложениям компьютеров сети. Кроме того, можно ограничить интенсивность сообщений о недостижимости порта назначения компьютеров сети.

UDP/echo/chargen-затопление

Атака **UDP/echo/chargen-затоплением** похожа на предыдущую, в ней также имеет место отражение UDP-пакетов, но при этом пакеты отправляются с номером порта 7 или 19. Эти порты обычно активны, они поддерживают сервисы echo (порт 7) и chargen (порт 19), использующие протокол UDP. Сервис chargen в ответ на запрос генерирует строку случайных символов случайной длины от 0 до 512 и посылает ее обратившемуся хосту. Этот сервис был встроен в ОС Unix для отладки ее сетевых функций. Аналогичное назначение имеет сервис echo (не путать с эхо-запросами и эхо-ответами протокола ICMP), он просто возвращает строку любого запроса по адресу обратившегося хоста.

В простейшем случае атакующий посылает UDP-пакеты на порт 7 и/или 19 некоторого промежуточного хоста и указывает обратный адрес атакуемого хоста. Промежуточный хост начинает бомбардировать атакуемый хост ответами сервисов chargen и/или echo. Правда, усиления атаки не происходит, так как размер ответов невелик; для усиления можно использовать широковещательный адрес промежуточной сети. Более интересной выглядит атака, когда атакующий посылает пакет с портом 19 и указывает в нем исходный порт 7. В этом случае единственный пакет атакующего вызывает бесконечный обмен пакетами между сервисом chargen промежуточного хоста и сервисом echo атакуемого хоста.

IP-атаки

Протокол IP сам по себе не предоставляет злоумышленникам особых шансов для атак, так как работает без установления соединения и достаточно прост в реализации. Тем не менее некоторые возможности для атак существуют.

Атака на IP-опции

Эта атака представляет собой DoS-атаку на маршрутизаторы, в которой используется поле дополнительных опций протокола IP.

В *IPv4* заголовок IP-пакета может включать поле опций, которые задают некоторую *нестандартную обработку* пакета маршрутизатором. Например, существует опция строгой маршрутизации от источника, которая позволяет отправителю IP-пакета задать точный список адресов промежуточных маршрутизаторов, через которые должен проходить маршрут доставки пакета, в то время как опция свободной маршрутизации от источника задает только некоторые из промежуточных маршрутизаторов маршрута. Опция фиксации маршрута требует от маршрутизаторов фиксации в пакете адресов промежуточных

маршрутизаторов, которые передавали пакет. Существует также возможность для производителей маршрутизаторов определять свои типы опций.

Атака основана на том факте, что у большинства IP-пакетов поле опций отсутствует, поэтому для продвижения таких пакетов маршрутизатор задействует специализированные процессоры портов, которые очень быстро и экономно выполняют эту операцию. А вот если встречается пакет с полем опций, то специализированный процессор его обработать не может и передает пакет центральному процессору маршрутизатора, в результате обработка пакета существенно замедляется. Поэтому поток пакетов, у которых присутствует одна или несколько опций, может привести к серьезному замедлению работы маршрутизатора, в предельном случае — к отказам в обслуживании нормальных пакетов. Усугубляет ситуацию присутствие в пакете двух взаимоисключающих опций, например строгой маршрутизации от источника и свободной маршрутизации от источника с разными промежуточными адресами.

Спецификация *IPv6* допускает наличие нескольких заголовков в пакете — основного и нескольких дополнительных. Вместо полей опций в пакете IPv6 могут присутствовать дополнительные заголовки, одним из которых является заголовок пошаговых опций (Hop-by-hop Options). Как и в случае опций IPv4, опции дополнительного заголовка пошаговых опций IPv6 обрабатываются центральным процессором маршрутизатора. Помещение в такой заголовок большого числа опций неопределенного типа будет замедлять работу маршрутизатора IPv6.

Обычная практика борьбы с этой атакой — фильтрация (отбрасывание) всех пакетов, в заголовке которых имеются опции. Возможно также игнорирование всех или некоторых опций.

IP-атака на фрагментацию

Атака на фрагментацию направлена на конечные узлы IP-сетей, в обязанность которых входит сборка фрагментированного IP-пакета в единое целое. Как оказалось, операция сборки имеет несколько уязвимостей, которые могут быть использованы злоумышленником:

- *Превышение максимальной длины пакета* (переполнение буфера сборки). Этот способ атаки уже был упомянут при описании атаки Пинг смерти. Максимальное значение смещения фрагмента равно $(2^{13} - 1) \times 8 = 8191 \times 8 = 65\,528$. Так как максимальная длина IP-пакета равна 65 535 байт, очевидно, что последний фрагмент не должен иметь длину более 7 байт. Задавая фрагмент с максимальным смещением и размером в восемь и более байтов, злоумышленник переполняет буфер ядра ОС, что может привести к падению ОС.
- *Перекрытие сегментов за счет специального подбора смещений и длин фрагментов.* Некоторые ОС не справляются со сборкой таких пакетов и падают. Например, эта уязвимость используется в атаке Teardrop.
- *Замещение фрагментов.* Эта DoS-атака используется для обмана таких защитных средств, как файерволы и системы обнаружения вторжений. Пакеты атаки фрагментируются и посылаются вместе с фрагментами-дубликатами, в которых содержится безобидная информация. Первым посылается безобидный фрагмент, а потом — фрагмент, содержащий код атаки, но с такими же смещением и длиной. В результате фрагмент

атаки замещает безобидный фрагмент. Не все файерволы и системы обнаружения вторжений распознают фрагментированную таким образом атаку.

- *Незавершенные фрагменты.* Эта DoS-атака направлена на исчерпание буферов сборки фрагментов. Атакующий посылает большое количество маленьких фрагментов, по паре на каждый собираемый пакет. Первый фрагмент из пары посылается с нулевым смещением, второй — с максимальным, поэтому они занимают максимальный объем памяти, отводимый под буфер. При отправке большого числа таких фрагментов за время тайм-аута сборки вся память ядра ОС, отводимая под сборку пакетов, оказывается исчерпанной, в результате наступает отказ в обслуживании фрагментированных пакетов.

Атаки на фрагментацию протокола IPv6 в принципе аналогичны атакам на фрагментацию IPv4, но так как в новой версии протокола IP был учтен опыт предыдущей борьбы, у злоумышленника остается меньше вариантов. Например, хотя превышение максимальной длины результирующего сегмента по-прежнему возможно, современные ОС при использовании протокола IPv6 предотвращают такую возможность.

Сетевая разведка

Задачи и разновидности сетевой разведки

Как можно увидеть из описания атак на сетевую транспортную инфраструктуру, многие из них требуют предварительных знаний об атакуемой сети и ее хостах. Например, для проведения ICMP-атаки Smurf нужно найти промежуточную сеть с большим количеством хостов, отвечающих на эхо-запросы, при этом такая сеть должна быть достижима для пакетов с широковещательным адресом этой сети, посланных из сети злоумышленника; ну и безусловно, должен быть известен IP- адрес атакуемого компьютера. Для атаки ICMP/UDP-затоплением нужно знать адреса хостов промежуточной сети, а также номера пассивных портов этих хостов; для атаки ICMP/chargen/echo-затоплением — адреса хостов промежуточной сети с активными портами 7 и 19, и т. д.

Если злоумышленник хочет задействовать сеть ботов, зараженных вирусом определенного типа, то ему понадобится просканировать большое количество компьютеров на отклик по определенному порту, который используется этим вирусом для получения команд от контроллера атаки. Дело в том, что вирусы стараются распространиться на возможно большее число компьютеров, но заранее нельзя сказать, будет ли успешным такое внедрение для какого-то определенного хоста, — это зависит от конфигурации средств защиты и других параметров ОС хоста. Поэтому злоумышленник заранее не знает, какие хосты он может использовать в качестве членов сети ботов, даже если это он инициировал распространение этого вируса. А возможно, он просто решит воспользоваться известным вирусом, распространенным другими лицами, и поэтому ему нужно собрать сведения о зараженных компьютерах.

Поэтому почти любую атаку предваряет сетевая разведка, при которой злоумышленник пытается собрать необходимые для атаки сведения. Конкретный набор сведений зависит от типа атаки, но чаще всего сетевая разведка включает сбор следующих данных: IP-адреса активных (то есть включенных, отвечающих на сетевой трафик) хостов; номера активных TCP-портов; номера активных и пассивных UDP-портов хостов; тип и версии ОС и приложений.

Сканирование сети

Обнаружение IP-адресов активных хостов сети называют **сканированием сети** (network scanning), а активных и пассивных портов — **сканированием портов** (port scanning). Сам термин «сканирование» говорит о том, что злоумышленник тестирует один за другим все возможные значения IP-адресов некоторой подсети (например, для подсети с маской /24 это 254 значения) или номера портов (65 535 для TCP и столько же для UDP).

Для сканирования сети и портов используются более изощренные средства, нежели утилита ping или стандартная процедура установления TCP-соединения, которые легко блокируются файерволами.

Перечислим наиболее распространенные приемы сканирования сети.

- **Пинг**[1] **TCP SYN** к одному из публично доступных портов, чаще всего к порту 80 (порт веб-сервера), который с большой степенью вероятности (но, конечно, не обязательно) открыт для внешнего доступа. Если хост отвечает пакетом SYN/ACK, то сканер считает, что *хост активен*, и завершает TCP-соединение пакетом с признаком RST.
- **Пинг TCP ACK** позволяет во многих случаях обойти файервол, если тот блокирует выбранный порт. Обычной практикой конфигурирования файервола является разрешение трафика *уже установленных* TCP-соединений, а признаком принадлежности пакета к такому соединению является наличие установленного флага ACK. В том случае, когда пинг TCP SYN к некоторому порту не проходит, а TCP ACK проходит, сканер считает, что данный *хост активен, но защищен файерволом*, — такая информация может быть ценной для злоумышленника.
- **Пинг UDP**. На тестируемый хост направляется UDP-пакет с номером порта, который, как рассчитывает злоумышленник, с большой степенью вероятности является *пассивным*. В том случае, когда компьютер включен и этот порт пассивен, сканер получает в ответ ICMP-сообщение о недоступности порта; если же компьютер отключен, то злоумышленник получает сообщение от маршрутизатора о недоступности хоста.
- **Пинг ICMP**. Администраторы сетей чаще всего блокируют эхо-запросы протокола ICMP, однако для проверки активности хоста злоумышленник может использовать *другие типы ICMP-запросов*, например запрос о длине маски IP-адреса (код 17) или запрос синхронизации времени протокола ICMP (код 13).
- **Пинг IP**. На исследуемый компьютер направляется IP-пакет с кодом протокола, отличным от кодов протоколов TCP, UDP и ICMP. Скорее всего, такой тип протокола не поддерживается стеком TCP/IP данного компьютера, и в том случае, если хост активен, в ответ будет послано ICMP-сообщение о недостижимости протокола.

Сканирование портов

Очень похожие методы применяются и для сканирования портов. Здесь предпочтение отдается SYN-сканированию протокола TCP, так как это самый быстрый способ, что в данном случае имеет значение, ведь в отличие от сканирования хостов здесь проверяют-

---

[1] В этом перечне процедур термин «пинг» использован в широком смысле, он не указывает конкретно на утилиту ping, работающую по протоколу ICMP, а говорит о том, что данные процедуры подобно утилите ping тестируют активность хоста с определенным IP-адресом.

ся десятки тысяч портов (по 65 535 портов для TCP и UDP). Сканирование портов часто осуществляется с помощью тех же специализированных программных средств, что и для инвентаризации сети и аудита ее защищенности. (Процедуры при этом используются те же, только цели их применения отличаются.)

Сканирование сети и портов обычно не проходит незамеченным — очень вероятно, что средства протоколирования событий ОС и файерволов зафиксируют этот процесс, и администратор сканируемой сети начнет расследовать инцидент. И первый вопрос, возникающий при этом: с какого адреса выполнялось сканирование? Чтобы избежать раскрытия, злоумышленники часто используют спуфинг IP-адреса при атаках. На первый взгляд кажется, что в сетевой разведке этот прием не может сработать, так как злоумышленнику нужно получать ответы на свой компьютер, иначе он не может получать информацию. Тем не менее спуфинг IP-адреса возможен и при сканировании. Одним из приемов является маскировка его среди множества других адресов. В этом случае тестовые сканирующие пакеты отправляются с действительного IP-адреса наряду с множеством таких же пакетов, но с поддельными адресами. Расчет здесь на то, что при расследовании факта сканирования трудно будет установить, кто являлся истинным организатором сканирования, а кого просто использовали как прикрытие. Еще более изощренным является так называемое пустое сканирование, когда истинный адрес никогда не указывается, а результаты сканирования злоумышленники пытаются понять по реакции третьего компьютера, чей адрес подделывается.

Атаки на DNS

Центральная роль службы DNS делает ее, с одной стороны, желанной целью атакующего, поскольку нарушение работы DNS наносит огромный ущерб работе сети, а с другой — мощным средством для проведения атак на другие сетевые механизмы, потому что многие из них оказываются безоружными перед этой глобальной службой.

DNS-спуффинг

В этой атаке DNS является не целью, а средством (рис. 29.5). Предположим, злоумышленник пытается получить доступ к корпоративному серверу www.example.com. Для этого ему нужны аутентификационные данные какого-нибудь его клиента.

Он решает перенаправить поток данных, которые легальный корпоративный клиент посылает корпоративному серверу, на свой компьютер. Для этого нужно опередить ответ DNS-сервера резольверу клиента и навязать ему свой вариант ответа, в котором вместо IP-адреса корпоративного сервера (в примере — 193.25.34.125) злоумышленник указывает IP-адрес атакующего хоста (203.13.1.123). На пути реализации этого плана имеется несколько серьезных препятствий.

Прежде всего, необходимо задержать ответ DNS-сервера, например, подвергнув его DoS-атаке. Другая проблема связана с определением номера порта DNS-клиента, который необходимо указать в заголовке пакета, чтобы данные дошли до приложения, так как если серверная часть DNS имеет постоянно закрепленный за ней так называемый хорошо известный номер порта 53, то клиентская часть протокола DNS получает номер порта динамически при запуске, причем операционная система выбирает его из достаточно широкого

Рис. 29.5. Схема перенаправления трафика путем использования ложных DNS-ответов

диапазона. Эту задачу злоумышленник решает путем прямого перебора всех возможных номеров. Также путем перебора возможных значений злоумышленник преодолевает проблему определения идентификаторов DNS-сообщений. Эти идентификаторы передаются в DNS-сообщениях и служат для того, чтобы DNS-клиент мог установить соответствие поступающих ответов посланным запросам. Итак, злоумышленник бомбардирует клиентскую машину ложными DNS-ответами, перебирая все возможные значения идентифицирующих полей так, чтобы клиент в конце концов принял один из них за истинный DNS-ответ. Как только это происходит, цель злоумышленника можно считать достигнутой — пакеты от клиента направляются на адрес атакующего хоста, злоумышленник получает в свое распоряжение имя и пароль легального пользователя, а с ними и доступ к корпоративному серверу.

Отравление кэша DNS

Атака **отравлением кэша DNS** направлена на замену корректной записи, хранящейся в кэше некоторого DNS-сервера, поддельной записью, которая направляет DNS-клиента на ложный узел. Эта атака более эффективна, чем DNS-спуфинг, так как, будучи успешной, она воздействует на большое количество клиентов в течение длительного времени (времени жизни подложной записи в кэше).

Атаку отравлением кэша DNS иллюстрирует рис. 29.6. Первым шагом злоумышленника является направление на атакуемый DNS-сервер ns.victim.org запроса на несуществующее имя, относящееся к домену злоумышленника. Получив такой запрос и не имея на него от-

вета в кэше (так как данного имени не существует), DNS-сервер ns.victim.org пересылает запрос серверу ns.hacker.org, который ведет зону hacker.org (найдя этот сервер через обычную процедуру поиска, начинающуюся с запроса к корневому серверу).

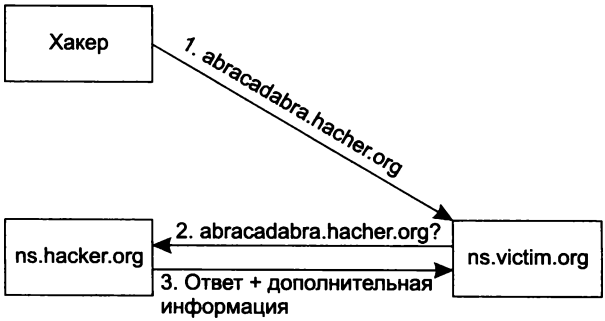


Рис. 29.6. Общая схема отравления DNS-кэша

DNS-сервер зоны ns.hacker.org находится под контролем хакера. В ответ на запрос несуществующего имени он отправляет особым образом сформированный ответ. Основное поле ответа, куда помещается запись о соответствии имени IP-адресу, обычно оставляется пустым (оно в атаке не используется), а вот в поле дополнительной информации хакер помещает запись, которая и должна отравить сервер ns.victim.org. Например, это может быть запись типа A о некотором хосте, который не принадлежит к зоне victim.org, поэтому сервер его кэширует и в дальнейшем использует при запросах своих клиентов. Это может быть и запись типа NS, в которой вместо имени легитимного DNS-сервера некоторого домена xxx.com указано имя DNS-сервера хакера, — в этом случае все запросы клиентов сервера ns.victim.org к хостам домена xxx.com будут направляться DNS-серверу хакера.

Атаки отравлением DNS-кэша носят исторический характер, так как в современных программных реализациях BIND DNS-сервер не кэширует информацию, если она непосредственно не относится к запросу (как в данном примере) или выходит за пределы зоны, для которой отвечающий сервер является полномочным.

Атаки на корневые DNS-серверы

Наиболее мощными и ощутимыми по своим последствиям были две DDoS-атаки на корневые серверы, случившиеся 21 октября 2002 года и 6/7 февраля 2007 года.

Подробности **атаки 21 октября 2002 года** приводятся в отчете специалистов, администрирующих корневые серверы[1]:

- Атака длилась чуть больше часа и была направлена на все 13 адресов корневых серверов.
- Атака была комбинированной, использовались ICMP-атака ping-затоплением, TCP-атака затоплением SYN-пакетами, атака фрагментированными IP-пакетами и атака UDP-затоплением.

[1] http://d.root-servers.org/october21.txt

- Интенсивность атаки на сервер — 50–100 Мбит/с; суммарная интенсивность — 900 Мбит/с.
- В атаке использовался спуфинг IP-адресов, отследить реальные источники атаки не удалось.

Служба DNS показала хорошую устойчивость к атаке — пользователи замечали только небольшое увеличение времени ожидания при открытии сайта в браузере; все корневые серверы продолжали работать, и на все принятые ими запросы были выданы ответы, но из-за перегрузки входных интерфейсов некоторых серверов не все запросы были приняты. После этой атаки были проведены дополнительные работы по повышению устойчивости службы DNS, которые включали повышение скорости интерфейсов и линий связи, соединяющих корневые серверы с Интернетом, и увеличение числа корневых серверов. Кроме того, корневые серверы были более равномерно распределены по автономным системам и географическим регионам.

Атака 6/7 февраля 2007 года длилась 24 часа (поэтому в названии фигурируют две даты). Эта атака была намного мощнее, чем атака 21 октября 2002 года, интенсивность трафика достигала 1 Гбит/с на один пул корневых серверов, но атаковано было только четыре из них. В атаке было использовано 4500–5000 компьютеров под управлением Microsoft Windows, причем члены этой сети ботов были распределены по сетям нескольких стран. При атаке имело место затопление корневых серверов UDP-пакетами, направленными на порт 53 (порт DNS), то есть атака относилась к типу UDP-затопления, а использование порта 53 помогало пакетам добраться до серверов, так как у файерволов, защищающих DNS-серверы, этот порт всегда открыт, иначе сервер не смог бы выполнять свою работу. Атака привела к почти полному исчерпанию пропускной способности двух из четырех атакованных пулов серверов, в то время как остальные два пострадали не так существенно и могли отвечать на большую часть запросов.

Атака практически немедленно была обнаружена центрами, ответственными за администрирование атакованных пулов корневых серверов (по предупреждающим сообщениям самих серверов и данным хостов, выполняющих постоянный мониторинг корневых серверов путем отправки на них контрольных запросов). Для снижения эффекта атаки было предпринято ряд мер, первой из которых была блокировка любых DNS-запросов, длина которых превышала 300 байт, так как обычно DNS-запрос не больше 100 байт, а в атакующих сообщениях для усиления эффекта затопления размеры полей данных UDP-потока доходили до 1023 байт. Однако такая блокировка помогла только частично, так как последующий анализ показал, что размер поля данных трафика атаки менялся случайным образом от 0 до 1023 байт. Анализ атаки также показал, что атакующие компьютеры не использовали спуфинг IP-адресов, что дало возможность отследить размещение ботов (в процентном отношении): Южная Корея — 65 %, США — 19 %, Канада — 3,5 %, Китай — 2,5 %, остальные страны — 10 %. Хост, координирующий атаку, находился в США, хосты сети ботов обращались к нему по протоколу HTTP.

Причины атаки так и остались неясными; в отчете ICANN предполагается, что это могло быть просто тщеславие хакеров, так как попытка остановить весь Интернет является вызовом для любого хакера.

Мы рассмотрели эти две атаки, потому что они дают хорошее представление о масштабах современных DDoS-атак и о том, что защититься от них очень сложно даже таким опытным специалистам, которые обслуживают корневые DNS-серверы. В то же время мы

видим, что такое архитектурное решение, как виртуализация серверов, когда логический сервер представлен большим пулом физических серверов, рассредоточенных по разным сетям и автономным системам, является очень мощным фактором, гасящим эффект даже очень интенсивной DDoS-атаки. В эффективности такого подхода мы еще раз убедимся в главе 30 при рассмотрении безопасности облачных вычислений.

DDoS-атаки отражением от DNS-серверов

Основная идея атаки отражения в данном случае состоит в следующем. В Интернете работают миллионы DNS-серверов, основной обязанностью которых является отправка ответов на запросы клиентов. При этом ответ может по объему намного превосходить запрос: например, в том случае, если запрос относится к передаче файла зоны (запрос типа AXFR) и зона включает большое количество записей.

Рассмотрим схему атаки такого рода на примере атаки[1], которой в марте 2013 года подвергся веб-сервер компании Spamhouse — некоммерческой организации, борющейся со спамом. Общая схема атаки изображена на рис. 29.7. Для организации этой длительной атаки было использовано около 30 000 DNS-серверов, работающих в открытом рекурсивном режиме, то есть отвечающих на запросы любых пользователей и при этом дающих полный (рекурсивный) ответ.

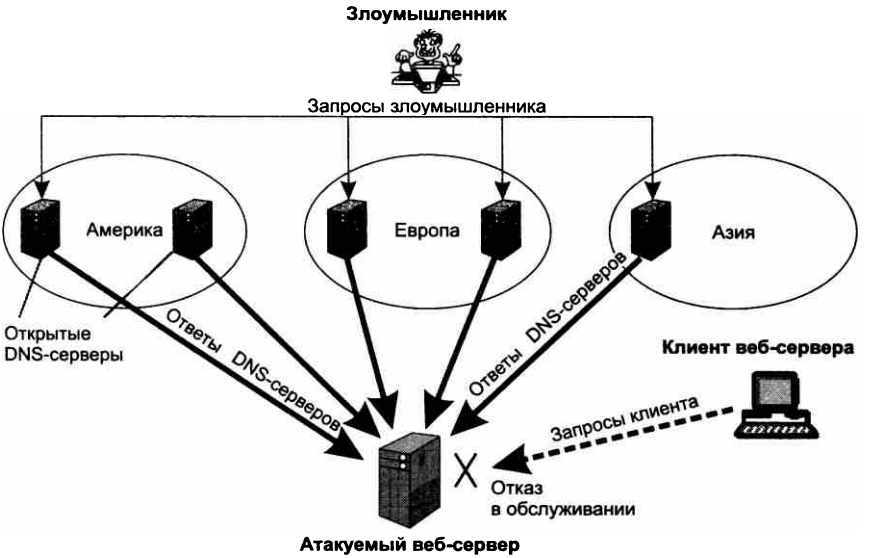


Рис. 29.7. Атака отражением от DNS-серверов

Рекурсивный режим здесь является важным элементом атаки, так как нерекурсивные DNS-серверы только перенаправляют запрашивающего на другой DNS-сервер, поэтому их ответ является коротким и не может усилить атаку. Общей практикой является поддержание

[1] http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho.

рекурсивного режима ответов только для «своих» клиентов — сотрудников предприятия для корпоративного DNS-сервера или же подписчиков сервиса для интернет-провайдера. Тем не менее в Интернете работает около 28 миллионов открытых рекурсивных DNS-серверов, так что найти такие серверы для атаки не так уж трудно.

Злоумышленником был послан поток запросов на 30 000 открытых DNS-серверов. В запросах в качестве адреса отправителя, как всегда это делается в атаках с отражением, злоумышленник указывал адрес атакуемого веб-сервера. Так что все ответы от 30 000 DNS-серверов обрушились на веб-сервер компании Spamhouse.

Для усиления атаки использовались не обычные запросы на разрешение имени, а запросы на передачу объемного файла со всеми записями зоны ripe.net (RIPE NCC — это региональный информационный интернет-центр по Европе). Файл зоны ripe.net имеет размер около 3000 байт, так что при размере запроса в 28 байт коэффициент усиления составил около 100. Такое мощное усиление позволило создать атаку с общей интенсивностью в 75 Гбит/с, используя поток запросов к одному DNS-серверу интенсивностью всего в 2,5 Мбит/с. Для отдельного DNS-сервера такой поток запросов не является чем-то необычным, так что владельцы этих серверов, скорее всего, эту атаку не заметили, а вот результирующий поток атаки в 75 Гбит/с вывел веб-сервер компании Spamhouse из строя.

Точнее, веб-сервер Spamhouse был выведен из строя только до определенного момента, пока его владельцы не перевели его «под крыло» CloudFlare — провайдера облачных сервисов, к тому же специализирующегося на защите от DDoS-атак. Перевод помог, так как распределенная виртуальная структура CloudFlare, использующая технику anycast и файерволы, смогла абсорбировать большую часть трафика атаки, и веб-сервер Spamhouse вновь стал доступен пользователям Интернета.

Методы защиты службы DNS

Существует ряд мер предосторожности, которые повышают защищенность DNS-серверов от атак или использования их в качестве инструмента атаки.

- *Защита ОС хоста.* Так как DNS — это приложение ОС, то сама ОС должна быть надежно защищена всеми возможными способами.
- *Разделение пользователей на внутренних и внешних.* Рекурсивные неполномочные ответы должны предоставляться только внутренним пользователям как вызывающим большее доверие.
- *Передача файла зоны из первичного сервера только вторичным серверам этой зоны* с использованием для передачи защищенных протоколов, например SFTP или SCP.
- *Использование DNSSEC.* **DNSSEC** представляет собой набор стандартов, обеспечивающих аутентификацию ответов DNS-серверов с помощью цифровой подписи и системы публичных ключей. DNSSEC-клиент может проверить, что полученный ответ действительно пришел от полномочного сервера зоны, а не от сервера, который просто утверждает, что он полномочен, а на самом деле может таковым и не являться. DNSSEC затрудняет злоумышленникам спуфинг-атаки и отравление кэша, так как для этого требуется подделывать цифровую подпись сервера. С 2010 года все корневые серверы, а также многие серверы верхнего уровня и крупных провайдеров поддерживают DNSSEC.

Безопасность маршрутизации на основе BGP

Протокол **BGP** (Border Gateway Protocol) в версии 4 является сегодня основным протоколом обмена маршрутной информацией между автономными системами Интернета. *Автономная система* — это совокупность сетей под единым административным управлением, в которой существует единая политика маршрутизации. Каждый провайдер Интернета управляет собственной автономной системой или несколькими автономными системами; крупные корпоративные сети также часто представляют собой отдельную автономную систему, более мелкие являются частью автономной системы провайдера. Автономные системы имеют номера. Разбиение Интернета на автономные системы позволяет декомпозировать процесс маршрутизации; внутри каждой автономной системы маршрутизация выполняется любым известным способом — с помощью статических маршрутов, то есть полностью вручную, или же на основе любого динамического протокола маршрутизации — это внутреннее дело организации, администрирующей данную автономную систему, главное, чтобы маршруты между узлами сетей автономной системы существовали и отвечали политике маршрутизации данной организации. Однако между автономными системами маршрутизация подчиняется общим правилам, которые реализованы в протоколе BGP.

Уязвимости и инциденты протокола BGP

Маршрутизация между автономными системам на основе протокола BGP является (наряду со службой DNS) одним из наиболее уязвимых элементов Интернета. Это объясняется, во-первых, тяжелыми последствиями, к которым приводит ошибочная работа BGP-маршрутизаторов сети провайдеров: маршруты ко многим частям Интернета вдруг исчезают или оказываются ложными для значительной части пользователей. Во-вторых, протокол BGP принципиально менее защищен, чем внутренние протоколы маршрутизации OSPF и IS-IS, так как «собеседники» BGP-маршрутизатора находятся за пределами административной ответственности его организации и поэтому возможностей для проверки достоверности маршрутных объявлений протокола BGP намного меньше, чем в случае внутренних протоколов.

Мы сознательно не использовали в заголовке этого раздела слово «атаки». Информация о случаях неправильной работы протокола BGP часто скрывается провайдерами Интернета, избегающими раскрытия деталей работы своей сети по разным причинам, в том числе и по причине безопасности. Поэтому большая часть крупных инцидентов, происшедших в Интернете по «вине» протокола BGP, остается инцидентами, а не атаками, так как трудно сказать, произошел тот или иной инцидент из-за ошибки конфигурирования маршрутизатора персоналом провайдера или же это была спланированная и осуществленная атака. Во многих статьях и документах, описывающих уязвимости этого протокола маршрутизации, появилось новое действующее лицо — *провайдер-злоумышленник* (malicious ISP), который вольно или невольно создает проблемы для остальных провайдеров.

Первый широко известный масштабный инцидент с BGP-маршрутизацией оказался весьма характерным и очень ярко выявил уязвимости BGP, которые затем много раз проявляли себя в аналогичных ситуациях. Этот инцидент произошел 25 апреля 1997 года, когда многие провайдеры обнаружили, что в их маршрутизаторах исчезли маршруты, описывающие путь к сетям Интернета. Причина этого неприятного обстоятельства довольно быстро

была найдена: в объявлениях автономной системы **AS7007** указывалось, что путь ко многим сетям Интернета должен пролегать через ее сети. Звонок провайдеру AS7007 помог устранить причину: оказалось, что виновником был единственный маршрутизатор одного из клиентов этого провайдера, который после реконфигурирования начал генерировать некорректные маршрутные объявления. Некорректность состояла в том, что в объявлениях указывались адреса, не относящиеся к пулу адресов данного провайдера, а кроме того, они являлись *более специфическими*, чем адреса в маршрутизаторах этого и большинства других провайдеров Интернета. После отключения виновного маршрутизатора таблицы маршрутизации провайдеров быстро восстановились, однако шок от того, как просто оказалось вывести Интернет из строя, остался надолго.

Детали этого и подобных инцидентов иллюстрирует рис. 29.8.

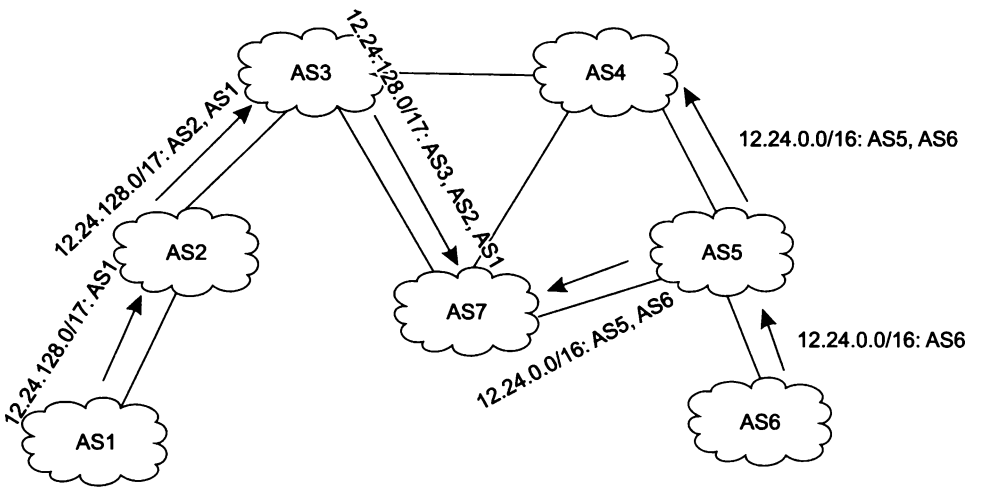


Рис. 29.8. Эффект деагрегирования адресов

В этом примере диапазон адресов 12.24.0.0/16 принадлежит AS1. BGP-маршрутизатор этой автономной системы объявляет о достижимости данных адресов через свою сеть. Маршрутизаторы автономной системы AS5 принимают это объявление, помещают запись о достижимости адресов 12.24.0.0/16 в свои таблицы маршрутизации и передают его далее маршрутизаторам автономных систем AS4 и AS7. Теперь посмотрим, что произойдет, если маршрутизаторы автономной системы AS1 начнут по ошибке или умышленно распространять маршрутные объявления к адресам диапазона 12.24.128.0/17. Этот диапазон является поддиапазоном диапазона 12.24.0.0/16, но его маска и, соответственно, префикс длиннее, а это и называется *более специфическим адресом*, и ему маршрутизатор должен отдавать предпочтение перед более коротким адресом. Поэтому маршрутизаторы AS7 поместят запись о том, что трафик к хостам с адресами из диапазона 12.24.128.0/17 должен направляться в AS1, а не в AS1. Эта информация распространится далее по всем автономным системам Интернета, в том числе по AS6, что приведет к потере связи с хостами 12.24.128.0/17, так как они находятся в AS6. Если система AS1 будет распространять только объявление 12.24.128.0/17, то остальные хосты из диапазона 12.24.0.0/16 окажутся достижимыми в AS6, если же AS1 будет распространять подобные объявления для

всех поддиапазонов этого диапазона с маской 17, таких как 12.24.0.0/17, 12.24.192.0/17, 12.24.224.0/17 и т. д., то все хосты сети 12.24.0.0/16 станут недостижимыми. Этот эффект называется **деагрегированием адресов**.

Манипуляции с маршрутными объявлениями

Инцидент с AS7007 был первой масштабной демонстрацией уязвимости маршрутизации на основе протокола BGP — протокола, который, как и другие протоколы стека TCP/IP, был разработан в расчете на добрую волю всех пользователей Интернета и не имел никакой защиты от ошибок или злого умысла. В дальнейшем подобные инциденты повторялись достаточно регулярно, один из них привлек большое внимание, потому что привел к временной недоступности популярного сервиса Youtube. Это случилось в 2008 году, когда оператор Pakistan Telecom пытался заблокировать доступ к Youtube для пользователей Пакистана, а вместо этого распространил всем провайдерам Интернета объявления о том, что специфические маршруты к Youtube ведут через его сеть, что привело к направлению мирового трафика Youtube в сеть Pakistan Telecom в течение двух часов.

Подобные инциденты являются следствием того, что маршрутное объявление протокола BGP формируется шаг за шагом многими провайдерами, при этом достоверность информации каждого шага проверить невозможно, так что у провайдера имеется полная свобода действий при обработке маршрутного объявления и передаче его соседним провайдерам. Например, вместо простого добавления номера своей автономной системы к уже имеющейся последовательности AS-номеров он может выполнить ряд следующих манипуляций с полученным маршрутом:

- Поместить адрес чужой сети с номером своей автономной системы в качестве исходной, чтобы направить трафик в свою автономную систему. Такая атака называется захватом префикса, и именно она произошла в инцидентах с AS7007 и Pakistan Telecom. Для того чтобы объявление адреса выглядело предпочтительнее, адрес должен быть более специфическим, чем у объявлений «настоящей» исходной автономной системы.
- Выбросить из последовательности какую-то определенную автономную систему, чтобы обойти политику некоей третьей автономной системы, которая по финансовым или иным соображениям блокирует все маршруты, проходящие через удаленную автономную систему.
- Добавить номер соседней автономной системы перед передачей ее объявления, чтобы соседняя автономная система, получив объявление и увидев в нем свой номер, отбросила его, решив, что объявление зациклилось.
- Добавить номер своей автономной системы несколько раз, чтобы объявление стало непривлекательным для других провайдеров (из-за размера последовательности AS).
- Составить ложную последовательность автономных систем, но поместить в качестве исходного правильный (но не свой) номер AS, чтобы вызвать доверие к маршруту.

Как видим, незащищенность маршрутного объявления дает большой простор для злонамеренных искажений и просто ошибок при его обработке. Вероятность ошибки усугубляется тем, что в отличие от внутренних протоколов маршрутизации, которые обрабатывают сообщения с минимальным вмешательством администратора, работа протокола BGP обычно регулируется большим количеством правил фильтрации, которые задаются вручную администратором AS. Эти фильтры определяют политику маршрутизации той или иной

автономной системы, отражающую взаимоотношения данного провайдера с каждым из провайдеров, с которым у него есть пиринговые соглашения о передаче трафика. Вместе с тем фильтры политики BGP представляют собой мощный и популярный способ защиты BGP-маршрутизации от ошибок и атак. Но для этого их нужно правильно применять.

Защита BGP

Первым заслоном на пути ошибок и атак типа «злоумышленник посередине» должна быть *защита BGP-сеанса между соседними маршрутизаторами*, особенно защита сеанса внешнего протокола BGP, так как маршрутизаторы в этом случае принадлежат разным провайдерам и между ними могут находиться промежуточные коммутаторы и другие маршрутизаторы, работающие не по протоколу BGP.

По умолчанию BGP-сеанс использует протокол TCP, поэтому описанные ранее атаки на TCP подвергают риску и работу BGP. Результатом атаки на TCP может стать удаление всех BGP-маршрутов из таблицы маршрутизации, так как все они могли быть получены в результате одного и того же длительного TCP-сеанса между BGP-маршрутизаторами. Подделка TCP-сегмента может привести к появлению ложного маршрута в таблице маршрутизации или удаления из нее корректного маршрута.

Для защиты TCP-сеанса между BGP-маршрутизаторами рекомендуется использовать режим работы TCP с аутентификацией сегментов посредством цифровой подписи.

Защита BGP-маршрутизации на основе базы данных маршрутов

С середины 90-х годов региональные информационные центры Интернета, которые распределяют IP-адреса и номера автономных систем среди провайдеров своих регионов (то есть RIPE NCC, ARIN, APNIC, LACNIC и AfriNIC), начали вести базу данных маршрутов Интернета.

В **базе данных маршрутов Интернета** (Internet Routing Registry, **IRR**) для каждого *зарегистрированного* провайдера указываются номера администрируемых им автономных систем и политика маршрутизации, проводимая этим провайдером в пределах каждой из своих автономных систем по отношению ко всем ее соседним автономным системам, с которыми у него установлены пиринговые отношения.

Например, пусть провайдер ISP1 администрирует автономную систему AS1 и у него имеются пиринговые соглашения с AS2, AS3 и AS4. Тогда в базе IRR будет существовать объект типа `aut-num` с параметрами такого вида:

```
aut-num:AS1
aut-name:ISP1
import:from AS2 action pref=50; accept AS2
export: to AS2 announce AS1
import:from AS3 action pref=50: accept any
export:to AS3 announce AS1
import:from AS4 action pref=50: accept AS3
export:to AS4 announce AS1
address:XX XXXXX XXXX
phone:YY-YYYYY-YYYYY
```

Из приведенного списка видно, что политика маршрутизации системы AS1 состоит в том, что ее маршрутизаторы объявляют каждой из пиринговых автономных систем только о тех маршрутах, которые ведут к адресам ее собственных сетей (об этом говорит атрибут `announce AS1`). В свою очередь, автономная система AS1 также готова принимать от AS2 и AS4 только те маршруты, которые исходят от адресов их собственных сетей, а от AS3 она готова принимать любые маршруты. Скорее всего, AS2 и AS4 являются клиентами AS1, а AS3 — это магистральный провайдер, через которого происходит связь AS1 с остальными автономными системами Интернета. В параметрах атрибутов `export` и `import` можно использовать не только номера автономных систем, но и префиксы IP-адресов.

Кроме объектов `aut-num` в базе IRR существуют объекты типа `route`, которые говорят о том, какие адреса будет объявлять та или иная автономная система в своих маршрутных объявлениях. Адреса, указанные в объектах `route`, являются подмножеством адресов, выделенных провайдеру, поскольку некоторые из них могут быть еще не назначены реальным сетям, другие могут быть предназначены для внутренней маршрутизации.

Регистрация в базе IRR не является обязательной для провайдеров, но она желательна, а иногда и необходима, так как некоторые провайдеры отказываются устанавливать пиринговые отношения с провайдерами, не зарегистрированными в базе IRR. Базы IRR всех пяти региональных центров RIR идентичны. База IRR является открытой, любой пользователь Интернета может запросить сведения о любой автономной системе с помощью команды `whois`. Обычной практикой провайдера является построение фильтров политики протокола BGP на своих маршрутизаторах на основании данных о политике соседей, полученных из базы IRR. Существует также утилита IRRToolSet, которая автоматизирует этот процесс и транслирует правила политики, описанные в базе IRR, в язык BGP-фильтров определенного типа маршрутизаторов.

Обращаясь снова к инциденту с автономной системой AS7007, заметим, что ее администратор мог бы легко предотвратить распространение специфических префиксов из маршрутизатора своего клиента, если бы установил простой фильтр, принимающий от маршрутизатора клиента только префиксы адресов, которые были назначены данному клиенту провайдером AS7007. Пострадавшие провайдеры соседних с AS7007 автономных систем также могли бы построить свои фильтры соответствующим образом, если бы провайдер AS7007 зарегистрировал свои объекты в базе IRR.

Несмотря на то что база IRR существует уже много лет и большинство провайдеров регистрируют в ней свои правила политики маршрутизации, инциденты с захватом префиксов по-прежнему регулярно случаются. Это стало поводом к началу работ по созданию новой безопасной масштабируемой версии протокола BGP, координируемых рабочей группой IETF SIDR (Secure Inter-Domain Routing).

Сертификаты ресурсов и их использование для защиты BGP

В качестве средства защиты BGP группа SIDR решила использовать публичную систему сертификатов[1].

[1] См. раздел «Системы аутентификации и управления доступом операционных систем» в главе 27.

Главным назначением **системы сертификатов ресурсов** (Resource Public Key Infrustructure, **RPKI**) является удостоверение того факта, что некоторый провайдер владеет определенными номерами автономных систем и префиксами IP-адресов.

Например, если провайдер ISP1 имеет сертификат RPKI, то этот сертификат показывает, что провайдеру *в установленном порядке* выделены номера автономных систем AS1, AS2, ..., ASn и префиксы IP1, IP2, IP3, ..., IPm. Установленный порядок означает, что номера и адреса были выданы либо IANA (корневая организация, выделяющая номера и адреса в Интернете), либо пятью региональными интернет-центрами, либо провайдерами, получившими эти адреса от региональных центров. Провайдеров в этой иерархии обычно называют **локальными интернет-центрами** (Local Internet Register, **LIR**).

Система RPKI состоит, как и любая система PKI, из **центров сертификации** (Certificate Authority, **CA**), при этом каждая организация из иерархии IANA→RIRs→LIRs имеет свой центр сертификации, который выдает сертификаты по запросу нижестоящей организации. RPKI-сертификаты имеют дополнительные поля для номеров автономных систем и префиксов адресов, в остальном же это обычный сертификат, в котором содержится открытый ключ владельца сертификата и имя владельца.

Сертификат называется *сертификатом ресурса*, потому что он предназначен не для аутентификации владельца сертификата, а для его *авторизации* (то есть наделения правами), — сертификат свидетельствует, что владелец имеет законное право распоряжаться номерами автономных систем и префиксов адресов, например передавать или продавать префиксы, указывать их в маршрутных объявлениях как исходные и т. п. Сертификаты здесь представляют собой масштабируемое решение, не требующее хранения множества паролей для проверки законности владения номером или номерами автономных систем и префиксов адресов некоторой организацией, вместо этого производится проверка предъявленного сертификата вдоль не очень длинной иерархии сертификационных центров.

Однако сам по себе RPKI-сертификат не может свидетельствовать о достоверности номера исходной автономной системы в маршрутном объявлении протокола BGP, так как обладатель некоторого префикса может делегировать право на объявление этого префикса в качестве маршрута автономной системе вышестоящего провайдера или клиента; наконец, некоторые адреса провайдер может не объявлять вовсе, зарезервировав их для внутреннего использования.

Поэтому для проверки законности объявления некоторой автономной системы как исходной для определенного префикса в маршрутном объявлении рабочая группа SIDR предложила использовать новый тип объекта, название которого можно перевести как объект **авторизации источника маршрута** (Route Origination Authorisation, ROA). ROA содержит номер автономной системы и несколько префиксов IP-адресов, которые эта автономная система имеет право объявлять в BGP-маршрутах. Объект ROA создается и подписывается владельцем префиксов, указанных в этом объекте.

Провайдеры могут использовать базу данных объектов ROA двумя способами. Во-первых, они могут задействовать данные этих объектов так же, как данные объектов `aut-num` и `route` из базы IRR для построения фильтров маршрутизаторов. Отличие состоит в том, что объекты ROA снабжены цифровой подписью, которую можно проверить, а объекты базы IRR — нет. Во-вторых, провайдеры могут автоматизировать процесс проверки достоверности источника маршрута. Для этого каждому провайдеру необходимо создать свой

локальный кэш базы RPKI, к которому могут обращаться маршрутизаторы при проверке каждого маршрутного объявления протокола BGP.

Возможность проверки достоверности номера исходной автономной системы для префикса сети является важным шагом в повышении защищенности протокола BGP от ошибок и атак. Однако это только первый шаг в нужном направлении, так как он не исключает манипуляций с маршрутными объявлениями при передаче их от провайдера к провайдеру. Даже тот факт, что указанная в маршруте исходная автономная система имела право объявить маршрут к тому или иному префиксу (факт, проверенный с помощью ROA), не гарантирует того, что маршрут был сгенерирован данной автономной системой, — его вполне мог скомпоновать и провайдер-злоумышленник.

Поэтому следующим этапом должно стать появление средств, которые позволят маршрутизаторам «на лету» проверять достоверность всех звеньев маршрута. Таким протоколом является, по мнению специалистов из группы SIDR, протокол **BGPSEC**, который основан на цифровых подписях каждого провайдера, участвующего в пошаговом формировании маршрутного объявления протокола BGP. Получив объявление, маршрутизатор провайдера проверяет цифровые подписи предыдущих провайдеров, чьи автономные системы указаны в объявлении, а затем добавляет свою подпись, которая подписывает предыдущую версию объявления с добавленными номером автономной системы данного провайдера и номером автономной системы следующего шага. Добавление номера автономной системы следующего шага препятствует перехвату и незаконной передаче маршрутного объявления не по назначению, то есть срывает атаку «злоумышленник посередине». Пошаговое удостоверение маршрута гарантирует также, что объявление прошло именно тот путь через последовательность автономных систем, который указан в данном объявлении. Для проверок цифровой подписи используются сертификаты, выпущенные в рамках системы RPKI.

Этот способ защиты BGP является наиболее радикальным, так как он требует полной замены текущей версии BGP в маршрутизаторах провайдеров.

Технологии защищенного канала

Известно, что задачу защиты данных можно разделить на две подзадачи: защиту данных внутри компьютера и защиту данных в процессе их передачи от одного компьютера к другому. Для обеспечения безопасности данных при их передаче по публичным сетям используются различные технологии защищенного канала.

Технология защищенного канала обеспечивает защиту трафика между двумя точками в открытой транспортной сети, например в Интернете. Защищенный канал подразумевает выполнение трех основных функций:

- взаимная аутентификация абонентов при установлении соединения, которая может быть выполнена, например, путем обмена паролями;
- защита передаваемых по каналу сообщений от несанкционированного доступа, например путем шифрования;
- подтверждение целостности поступающих по каналу сообщений, например путем передачи одновременно с сообщением его дайджеста.

Способы образования защищенного канала

В зависимости от месторасположения программного обеспечения защищенного канала различают две схемы его образования:

- схема с конечными узлами, взаимодействующими через публичную сеть (рис. 29.9, *а*);
- схема с оборудованием поставщика услуг публичной сети, расположенным на границе между частной и публичной сетями (рис. 29.9, *б*).

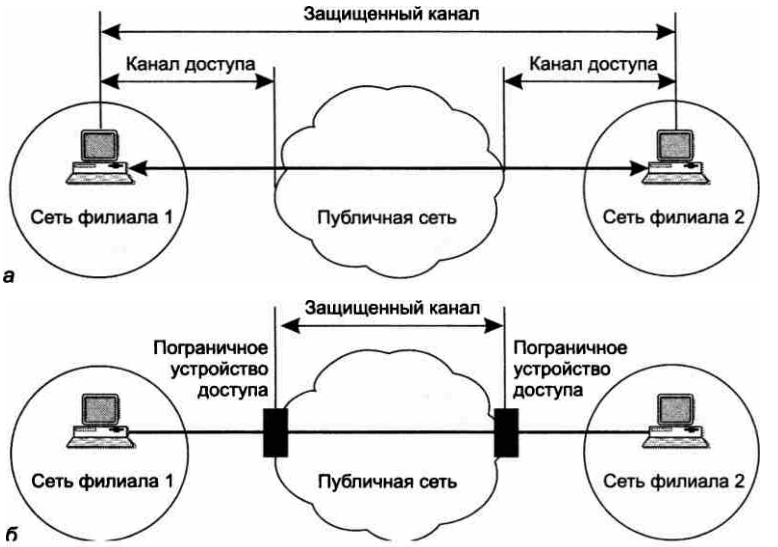


Рис. 29.9. Два подхода к образованию защищенного канала

В первом случае защищенный канал образуется программными средствами, установленными на двух удаленных компьютерах, принадлежащих двум разным локальным сетям одного предприятия и связанных между собой через публичную сеть. Преимуществом этого подхода является полная защищенность канала вдоль всего пути следования, а также возможность использования любых протоколов создания защищенных каналов, лишь бы на конечных точках канала поддерживался один и тот же протокол. Недостатки заключаются в избыточности и децентрализованности решения. Избыточность состоит в том, что вряд ли стоит создавать защищенный канал на всем пути следования данных: уязвимыми для злоумышленников обычно являются сети с коммутацией пакетов, а не каналы телефонной сети или выделенные каналы, через которые локальные сети подключены к территориальной сети. Поэтому защиту каналов доступа к публичной сети можно считать избыточной. Децентрализация заключается в том, что для каждого компьютера, которому требуется предоставить услуги защищенного канала, необходимо отдельно устанавливать, конфигурировать и администрировать программные средства защиты данных. Подключение каждого нового компьютера к защищенному каналу требует выполнять эти трудоемкие операции заново.

Во втором случае клиенты и серверы не участвуют в создании защищенного канала — он прокладывается только внутри публичной сети с коммутацией пакетов, например внутри

Интернета. Так, канал может быть проложен между сервером удаленного доступа поставщика услуг публичной сети и пограничным маршрутизатором корпоративной сети. Это хорошо масштабируемое решение, управляемое централизованно администраторами как корпоративной сети, так и сети поставщика услуг. Для компьютеров корпоративной сети канал прозрачен — программное обеспечение этих конечных узлов остается без изменений. Такой гибкий подход позволяет легко образовывать новые каналы защищенного взаимодействия между компьютерами независимо от места их расположения. Реализация этого подхода сложнее — нужен стандартный протокол образования защищенного канала, требуется установка у всех поставщиков услуг программного обеспечения, поддерживающего такой протокол, необходима поддержка протокола производителями пограничного коммуникационного оборудования. Однако вариант, когда все заботы по поддержанию защищенного канала берет на себя поставщик услуг публичной сети, оставляет сомнения в надежности защиты: во-первых, незащищенными оказываются каналы доступа к публичной сети, во-вторых, потребитель услуг чувствует себя в полной зависимости от надежности поставщика услуг.

Иерархия технологий защищенного канала

Защищенный канал можно построить с помощью системных средств, реализованных на разных уровнях модели OSI (рис. 29.10).

Рис. 29.10. Протоколы, формирующие защищенный канал на разных уровнях модели OSI

Если защита данных осуществляется средствами верхних уровней (прикладного, представления или сеансового), то такой способ защиты не зависит от технологий транспортировки данных (IP или IPX, Ethernet или ATM), что можно считать несомненным достоинством. В то же время приложения при этом становятся зависимыми от конкретного протокола защищенного канала, так как в них должны быть встроены явные вызовы функций этого протокола.

Защищенный канал, реализованный на самом высоком (прикладном) уровне, защищает только вполне определенную сетевую службу, например файловую, гипертекстовую или

почтовую. Так, протокол S/MIME защищает исключительно сообщения электронной почты. При таком подходе для каждой службы необходимо разрабатывать собственную защищенную версию протокола.

Популярный протокол **SSL** (Secure Socket Layer — слой защищенных сокетов) работает на уровне представления и создает защищенный канал, используя следующие технологии безопасности:

- взаимная аутентификация приложений на обоих концах защищенного канала выполняется путем обмена сертификатами (стандарт X.509);
- для контроля целостности передаваемых данных используются дайджесты;
- секретность обеспечивается шифрацией средствами симметричных ключей сеанса.

Протокол SSL разработан компанией Netscape Communications для защиты данных, передаваемых между веб-сервером и веб-браузером, но он может быть использован и любыми другими приложениями. Работа протокола защищенного канала на уровне представления делает его более универсальным средством, чем протокол безопасности прикладного уровня. Однако для того, чтобы приложение смогло воспользоваться протоколом уровня представления, в него по-прежнему приходится вносить исправления, хотя и не столь существенные, как в случае протокола прикладного уровня. Модификация приложения в данном случае сводится к встраиванию явных обращений к API соответствующего протокола безопасности.

Средства защищенного канала становятся прозрачными для приложений в тех случаях, когда безопасность обеспечивается на сетевом и канальном уровнях. Однако здесь мы сталкиваемся с другой проблемой — зависимостью сервиса защищенного канала от протокола нижнего уровня. Например, протокол PPTP, не являясь протоколом канального уровня, защищает кадры протокола PPP канального уровня, упаковывая их в IP-пакеты. При этом не имеет значения, пакет какого протокола, в свою очередь, упакован в данном PPP-кадре: IP, IPX, SNA или NetBIOS. С одной стороны, это делает сервис PPTP достаточно универсальным, так как клиент сервиса защищенного канала может задействовать любые протоколы в своей сети. С другой стороны, такая схема предъявляет жесткие требования к типу протокола канального уровня, используемому на участке доступа клиента к защищенному каналу, — для протокола PPTP таким протоколом может быть *только PPP*. Хотя протокол PPP очень распространен в линиях доступа, сегодня конкуренцию ему составляют протоколы Gigabit Ethernet и Fast Ethernet, которые все чаще работают не только в локальных, но и в глобальных сетях.

Работающий на сетевом уровне протокол IPSec является компромиссным вариантом. С одной стороны, он прозрачен для приложений, с другой — может работать практически во всех сетях, так как основан на широко распространенном протоколе IP и использует любую технологию канального уровня (PPP, Ethernet, ATM и т. д.).

Распределение функций между протоколами IPSec

Протокол IPSec в стандартах Интернета называют *системой*. Действительно, IPSec — это согласованный набор открытых стандартов, имеющий сегодня вполне очерченное ядро, которое в то же время может быть достаточно просто дополнено новыми функциями и протоколами.

Ядро IPSec составляют три протокола:

- AH (Authentication Header — заголовок аутентификации) гарантирует целостность и аутентичность данных;
- ESP (Encapsulating Security Payload — инкапсуляция зашифрованных данных) шифрует передаваемые данные, обеспечивая конфиденциальность, может также поддерживать аутентификацию и целостность данных;
- IKE (Internet Key Exchange — обмен ключами Интернета) решает вспомогательную задачу автоматического предоставления конечным точкам защищенного канала секретных ключей, необходимых для работы протоколов аутентификации и шифрования данных.

Как видно из краткого описания функций, возможности протоколов AH и ESP частично перекрываются (рис. 29.11). В то время как AH отвечает только за обеспечение целостности и аутентификации данных, ESP может шифровать данные и, кроме того, выполнять функции протокола AH (хотя, как мы увидим позднее, аутентификация и целостность обеспечиваются им в несколько урезанном виде). ESP может поддерживать функции шифрования и аутентификации/целостности в любых комбинациях, то есть либо всю группу функций, либо только аутентификацию/целостность, либо только шифрование.

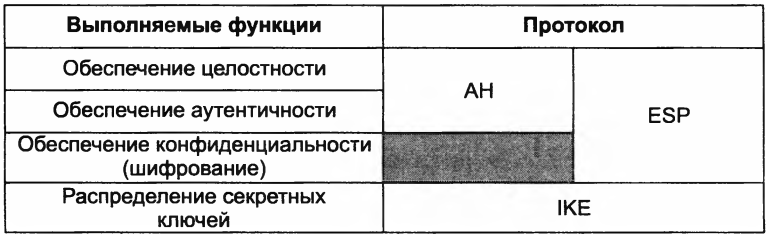


Рис. 29.11. Распределение функций между протоколами IPSec

Разделение функций защиты между протоколами AH и ESP вызвано применяемой во многих странах практикой ограничения экспорта и/или импорта средств, обеспечивающих конфиденциальность данных путем шифрования. Каждый из этих протоколов может использоваться как самостоятельно, так и одновременно с другим, так что в тех случаях, когда шифрование из-за действующих ограничений применять нельзя, систему можно поставлять только с протоколом AH. Естественно, подобная защита данных во многих случаях оказывается недостаточной. Принимающая сторона получает лишь возможность проверить, что данные были отправлены именно тем узлом, от которого они ожидаются, и дошли в том виде, в каком были отправлены. Однако от несанкционированного просмотра данных на пути их следования по сети протокол AH защитить не может, так как не шифрует их. Для шифрования данных необходим протокол ESP.

Безопасная ассоциация

Для того чтобы протоколы AH и ESP могли выполнять свою работу по защите передаваемых данных, протокол IKE устанавливает между двумя конечными точками логическое соединение (рис. 29.12), которое в стандартах IPSec носит название **безопасной ассоциации** (Security Association, SA).

Рис. 29.12. Безопасная ассоциация

Стандарты IPSec позволяют конечным точкам защищенного канала использовать единственную безопасную ассоциацию для передачи трафика всех взаимодействующих через этот канал хостов или создавать для этой цели произвольное число безопасных ассоциаций, например по одной на каждое TCP-соединение. Это дает возможность выбирать нужную степень детализации защиты — от одной общей ассоциации для трафика множества конечных узлов до индивидуально настроенных ассоциаций для защиты каждого приложения.

Безопасная ассоциация в протоколе IPSec представляет собой однонаправленное (симплексное) логическое соединение, поэтому если требуется обеспечить безопасный двусторонний обмен данными, необходимо установить две безопасные ассоциации. Эти ассоциации в общем случае могут иметь разные характеристики: например, в одну сторону при передаче запросов к базе данных достаточно только аутентификации, а для ответных данных, несущих ценную информацию, дополнительно можно обеспечить конфиденциальность.

Установление безопасной ассоциации начинается со взаимной аутентификации сторон, потому что все меры безопасности теряют смысл, если данные передаются или принимаются не тем лицом или не от того лица. Выбираемые далее параметры SA определяют, какой из двух протоколов, AH или ESP, будет применяться для защиты данных, какие функции будет выполнять протокол (например, можно выполнять только аутентификацию и проверку целостности, а можно, кроме того, еще и обеспечивать конфиденциальность). Очень важными параметрами безопасной ассоциации являются также секретные ключи, используемые в работе протоколов AH и ESP.

Протокол IPSec допускает как автоматическое, так и ручное установление безопасной ассоциации. При ручном способе администратор конфигурирует конечные узлы так, чтобы они поддерживали согласованные параметры ассоциации, включая секретные ключи. При автоматической процедуре установления SA протоколы IKE, работающие по разные стороны канала, выбирают параметры в ходе переговорного процесса. Для каждой задачи, решаемой протоколами AH и ESP, предлагается несколько схем аутентификации и шифрования (рис. 29.13). Это делает протокол IPSec очень гибким средством. Заметим, что выбор дайджест-функции для решения задач целостности и аутентификации никак не влияет на выбор функции шифрования, обеспечивающей конфиденциальность данных.

Для обеспечения совместимости в стандартной версии IPsec определен некоторый обязательный «инструментальный» набор; в частности, для аутентификации данных всегда может быть использована одна из стандартных дайджест-функций MD5 либо SHA-1, а в число алгоритмов шифрования непременно входит DES. При этом производители продуктов, в которых используется IPSec, вольны расширять протокол путем включения других алгоритмов аутентификации и симметричного шифрования, что они с успехом и делают. Например, многие реализации IPSec поддерживают популярный алгоритм шифрования Triple DES, а также сравнительно новые алгоритмы: Blowfish, Cast, CDMF, Idea, RC5.

Рис. 29.13. Согласование параметров в протоколе ESP

Транспортный и туннельный режимы

Протоколы AH и ESP могут защищать данные в двух режимах: транспортном и туннельном. В **транспортном режиме** передача IP-пакета через сеть выполняется с помощью оригинального заголовка этого пакета, а в **туннельном режиме** исходный пакет помещается в новый IP-пакет, и передача данных по сети выполняется на основании заголовка нового IP-пакета.

Применение того или иного режима зависит от требований, предъявляемых к защите данных, а также от роли, которую играет в сети узел, завершающий защищенный канал. Так, узел может быть хостом (конечным узлом) или шлюзом (промежуточным узлом). Соответственно имеются три схемы применения протокола IPSec:

- хост-хост;
- шлюз-шлюз;
- хост-шлюз.

В схеме хост—хост защищенный канал, или, что в данном контексте одно и то же, безопасная ассоциация, устанавливается между двумя конечными узлами сети (см. рис. 29.12). Тогда протокол IPSec работает на конечных узлах и защищает данные, передаваемые от хоста 1 к хосту 2. Для схемы хост—хост чаще всего используется транспортный режим защиты.

В соответствии со схемой шлюз—шлюз защищенный канал устанавливается между двумя промежуточными узлами, так называемыми **шлюзами безопасности** (Security Gateway, SG), на каждом из которых работает протокол IPSec (рис. 29.14). Защищенный обмен данными может происходить между любыми двумя конечными узлами, подключенными к сетям, которые расположены позади шлюзов безопасности. От конечных узлов поддержка протокола IPSec не требуется, они передают свой трафик в незащищенном виде через заслуживающие доверия внутренние сети предприятий. Трафик, направляемый в общедоступную сеть, проходит через шлюз безопасности, который и обеспечивает его защиту с помощью протокола IPSec. Шлюзам доступен только туннельный режим работы.

На рисунке пользователь компьютера с адресом IP1 посылает пакет по адресу IP2, используя туннельный режим протокола IPSec. Шлюз SG1 зашифровывает пакет целиком, вместе

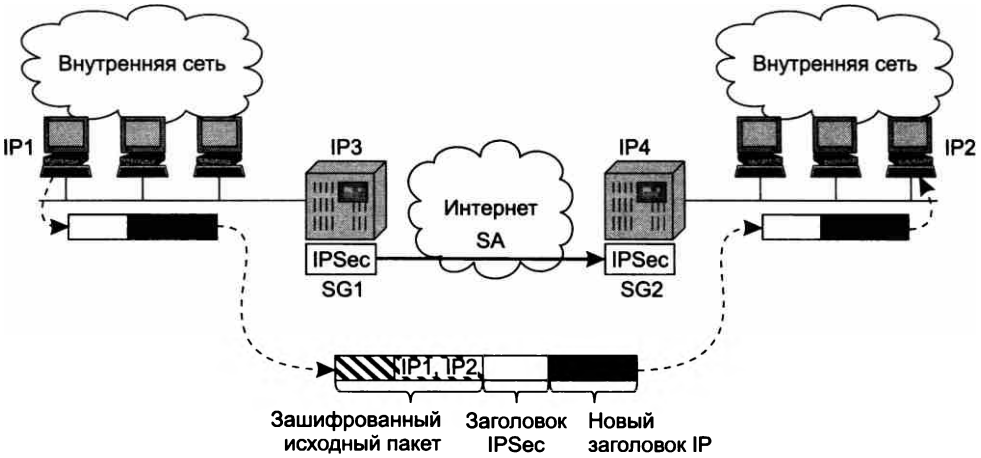


Рис. 29.14. Работа защищенного канала по схеме шлюз—шлюз в туннельном режиме

с заголовком, и снабжает его новым IP-заголовком, в котором в качестве адреса отправителя указывает свой адрес — IP3, а в качестве адреса получателя — адрес IP4 шлюза SG2. Вся передача данных по составной IP-сети выполняется на основании заголовка внешнего пакета, а внутренний пакет становится при этом полем данных для внешнего пакета. На шлюзе SG2 протокол IPSec извлекает инкапсулированный пакет и расшифровывает его, приводя к исходному виду.

Схема хост—шлюз часто применяется при удаленном доступе. В этом случае защищенный канал прокладывается между удаленным хостом, на котором работает протокол IPSec, и шлюзом, защищающим трафик для всех хостов, входящих во внутреннюю сеть предприятия. Эту схему можно усложнить, создав параллельно еще один защищенный канал — между удаленным хостом и каким-либо хостом, принадлежащим внутренней сети, защищаемой шлюзом (рис. 29.15). Такое комбинированное использование двух безопасных ассоциаций позволяет надежно защитить трафик во внутренней сети.

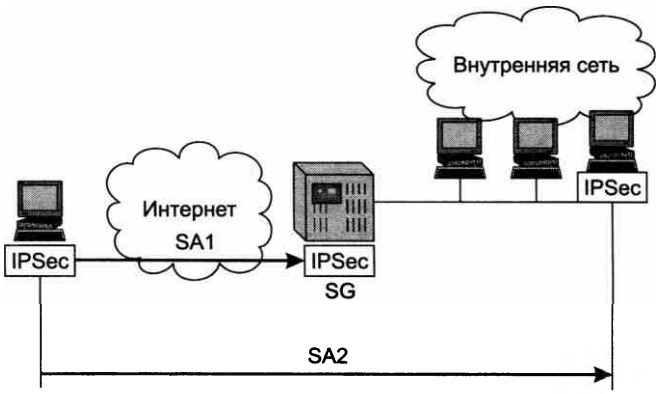


Рис. 29.15. Схема защищенного канала хост—шлюз

Протокол AH

Протокол AH позволяет приемной стороне убедиться, что:

- пакет был отправлен стороной, с которой установлена безопасная ассоциация;
- содержимое пакета не было искажено в процессе его передачи по сети;
- пакет не является дубликатом уже полученного пакета.

Две первые функции обязательны для протокола AH, а последняя выбирается по желанию при установлении ассоциации. Для выполнения этих функций протокол AH использует специальный заголовок (рис. 29.16).

Рис. 29.16. Структура заголовка протокола AH

В поле *следующего заголовка* (next header) указывается код протокола более высокого уровня, то есть протокола, сообщение которого размещено в поле данных IP-пакета. Скорее всего, им будет один из протоколов транспортного уровня (TCP или UDP) или протокол ICMP, но может встретиться и протокол ESP, если он используется в комбинации с AH.

В поле *длины полезной нагрузки* (payload length) содержится длина заголовка AH.

Индекс параметров безопасности (Security Parameters Index, SPI) служит для связи пакета с предусмотренной для него безопасной ассоциацией. Немного позже мы обсудим его более подробно.

Поле *порядкового номера* (Sequence Number, SN) указывает на порядковый номер пакета и применяется для защиты от его ложного воспроизведения (когда третья сторона пытается повторно использовать перехваченные защищенные пакеты, отправленные реально аутентифицированным отправителем). Отправляющая сторона последовательно увеличивает значение этого поля в каждом новом пакете, передаваемом в рамках данной ассоциации, так что приход дубликата обнаружится принимающей стороной (если, конечно, в рамках ассоциации будет активирована функция защиты от ложного воспроизведения). Однако в любом случае в функции протокола AH не входит восстановление утерянных и упорядочивание прибывающих пакетов — он просто отбрасывает пакет, когда обнаруживает, что аналогичный пакет уже получен. Чтобы сократить требуемую для работы протокола буферную память, используется механизм скользящего окна — на предмет дублирования проверяются только те пакеты, чей номер находится в пределах окна. Окно обычно выбирается размером в 32 или 64 пакета.

Поле *данных аутентификации* (authentication data), которое содержит так называемое **значение проверки целостности** (Integrity Check Value, ICV), служит для аутентификации и проверки целостности пакета. Это значение, называемое также дайджестом, вычисляется с помощью одной из двух обязательно поддерживаемых протоколом AH вычислительно

необратимых функций MD5 или SAH-1, но может использоваться и любая другая функция, о которой стороны договорились в ходе установления ассоциации. При вычислении дайджеста пакета в качестве параметра функции OWF выступает симметричный секретный ключ, который был задан для данной ассоциации вручную или автоматически с помощью протокола IKE. Так как длина дайджеста зависит от выбранной функции, это поле имеет в общем случае переменный размер.

Протокол AH старается охватить при вычислении дайджеста как можно большее число полей исходного IP-пакета, но некоторые из них в процессе передачи пакета по сети меняются непредсказуемым образом, поэтому не могут включаться в аутентифицируемую часть пакета. Например, целостность значения поля времени жизни (TTL) в приемной точке канала оценить нельзя, так как оно уменьшается на единицу каждым промежуточным маршрутизатором и никак не может совпадать с исходным.

Местоположение заголовка AH в пакете зависит от того, в каком режиме — транспортном или туннельном — сконфигурирован защищенный канал. Результирующий пакет в транспортном режиме выглядит так, как показано на рис. 29.17.

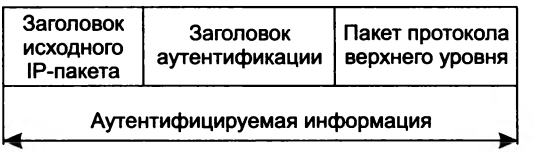


Рис. 29.17. Структура IP-пакета, обработанного протоколом AH в транспортном режиме

При использовании туннельного режима, когда шлюз IPSec принимает проходящий через него транзитом исходящий пакет и создает для него внешний IP-пакет, протокол AH защищает все поля исходного пакета, а также неизменяемые поля нового заголовка внешнего пакета (рис. 29.18).

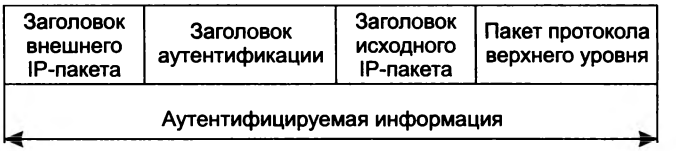


Рис. 29.18. Структура IP-пакета, обработанного протоколом AH в туннельном режиме

Протокол ESP

Протокол ESP решает две группы задач. К первой группе относятся задачи обеспечения аутентификации и целостности данных на основе дайджеста, аналогичные задачам протокола AH, ко второй — защита передаваемых данных путем их шифрования от несанкционированного просмотра.

Как видно на рис. 29.19, заголовок делится на две части, разделяемые полем данных. Первая часть, называемая собственно *заголовком ESP*, образуется двумя полями (SPI и SN), назначение которых аналогично одноименным полям протокола AH, и размещается перед

полем данных. Остальные служебные поля протокола ESP, называемые *концевиком ESP*, расположены в конце пакета.

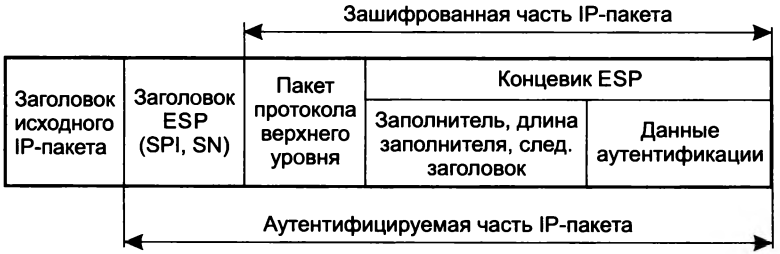


Рис. 29.19. Структура IP-пакета, обработанного протоколом ESP в транспортном режиме

Два поля концевика — *следующего заголовка* и *данных аутентификации* — также аналогичны полям заголовка AH. Поле данных аутентификации отсутствует, если при установлении безопасной ассоциации принято решение не использовать средства протокола ESP, касающиеся обеспечения целостности. Помимо этих полей концевик содержит два дополнительных поля — *заполнителя* и *длины заполнителя*. Заполнитель может понадобиться в трех случаях. Во-первых, для нормальной работы некоторых алгоритмов шифрования необходимо, чтобы шифруемый текст содержал кратное число блоков определенного размера. Во-вторых, формат заголовка ESP требует, чтобы поле данных заканчивалось на границе четырех байтов. И наконец, заполнитель можно использовать, чтобы скрыть действительный размер пакета в целях обеспечения так называемой частичной конфиденциальности трафика. Правда, возможность маскировки ограничивается сравнительно небольшим объемом заполнителя — 255 байт, поскольку большой объем избыточных данных может снизить полезную пропускную способность канала связи.

На рис. 29.19 показано размещение полей заголовка ESP в *транспортном режиме*. В этом режиме ESP не шифрует заголовок IP-пакета, иначе маршрутизатор не сможет прочитать поля заголовка и корректно осуществить продвижение пакета между сетями. В число шифруемых полей не попадают также поля SPI и SN, которые должны передаваться в открытом виде для того, чтобы прибывший пакет можно было отнести к определенной ассоциации и предотвратить ложное воспроизведение пакета.

В туннельном режиме заголовок исходного IP-пакета помещается после заголовка ESP и полностью попадает в число защищаемых полей, а заголовок внешнего IP-пакета протоколом ESP не защищается (рис. 29.20).

Рис. 29.20. Структура IP-пакета, обработанного протоколом ESP в туннельном режиме

Базы данных SAD и SPD

Итак, технология IPSec предлагает различные методы защиты трафика. Каким же образом протокол IPSec, работающий на хосте или на шлюзе, определяет способ защиты, который он должен применить к трафику? Решение основано на использовании в каждом узле, поддерживающем IPSec, двух типов баз данных:

- безопасных ассоциаций (Security Associations Database, SAD);
- политики безопасности (Security Policy Database, SPD).

При установлении безопасной ассоциации, как и при любом другом логическом соединении, две стороны принимают ряд соглашений, регламентирующих процесс передачи потока данных между ними. Соглашения фиксируются в виде набора параметров. Для безопасной ассоциации такими параметрами являются, в частности, тип и режим работы протокола защиты (AH или ESP), методы шифрования, секретные ключи, значение текущего номера пакета в ассоциации и другая информация. Наборы текущих параметров, определяющих все активные ассоциации, хранятся на обоих оконечных узлах защищенного канала в виде баз данных безопасных ассоциаций (SAD). Каждый узел IPSec поддерживает две базы SAD — одну для исходящих ассоциаций, другую для входящих.

Другой тип базы данных — база данных политики безопасности (SPD) — определяет соответствие между IP-пакетами и установленными для них правилами обработки. Записи SPD состоят из полей двух типов — полей селектора пакета и полей политики защиты для пакета с данным значением селектора (рис. 29.21).

Селектор в SPD включает следующий набор признаков, на основании которых можно с большой степенью детализации выделить защищаемый поток:

- IP-адреса источника и приемника могут быть представлены как в виде отдельных адресов (индивидуальных, групповых или широковещательных), так и диапазонами адресов, заданными с помощью верхней и нижней границ либо с помощью маски;
- порты источника и приемника (то есть TCP- или UDP-порты);
- тип протокола транспортного уровня (TCP, UDP);
- имя пользователя в формате DNS или X.500;
- имя системы (хоста, шлюза безопасности и т. п.) в формате DNS или X.500.

Для каждого нового пакета, поступающего в защищенный канал, IPSec просматривает все записи в базе SPD и сравнивает значение селекторов этих записей с соответствующими полями IP-пакета. Если значение полей совпадает с каким-либо селектором, то над пакетом выполняются действия, определенные в поле политики безопасности данной записи. Политика предусматривает передачу пакета без изменения, отбрасывание или обработку средствами IPSec.

В последнем случае поле политики защиты должно содержать ссылку на запись в базе данных SAD, в которую помещен набор параметров безопасной ассоциации для данного пакета (на рисунке для исходящего пакета определена ассоциация SA3). На основании заданных параметров безопасной ассоциации к пакету применяется соответствующие протокол (на рисунке — ESP), функции шифрования и секретные ключи.

Если к исходящему пакету нужно применить некоторую политику защиты, но указатель записи SPD показывает, что в настоящее время нет активной безопасной ассоциации с требуемой политикой, то IPSec создает новую ассоциацию с помощью протокола IKE, помещая новые записи в базы данных SAD и SPD.

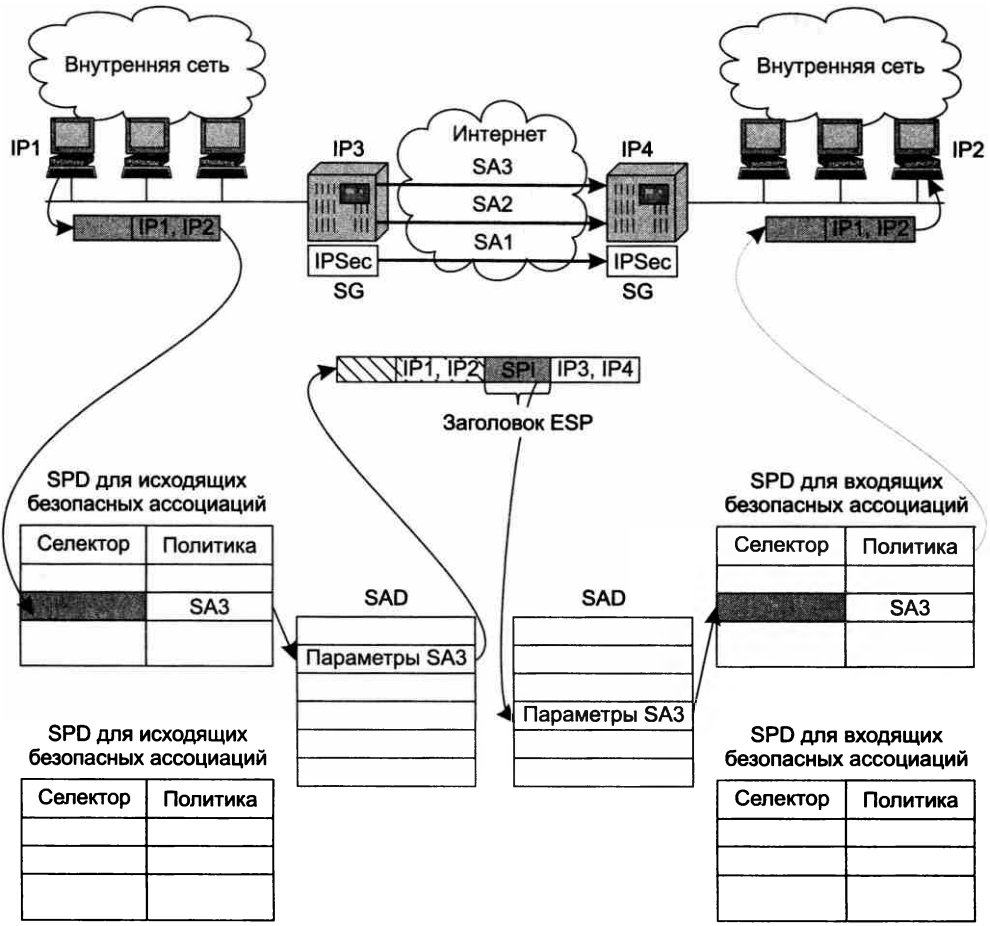


Рис. 29.21. Использование баз данных SPD и SAD

Базы данных политики безопасности создаются и администрируются либо пользователем (этот вариант больше подходит для хоста), либо системным администратором (вариант для шлюза), либо автоматически (приложением).

Ранее мы выяснили, что установление связи между исходящим IP-пакетом и заданной для него безопасной ассоциацией происходит путем селекции. Однако остается другой вопрос: как *принимающий* узел IPSec определяет способ обработки прибывшего пакета, ведь при шифровании многие ключевые параметры пакета, отраженные в селекторе, оказываются недоступными, а значит, невозможно определить соответствующую запись в базах данных SAD и SPD и, следовательно, тип процедуры, которую надо применить к поступившему пакету? Именно для решения этой проблемы в заголовках AH и ESP предусмотрено поле SPI. В это поле помещается указатель на ту строку базы данных SAD, в которой записаны параметры соответствующей безопасной ассоциации. Поле SPI заполняется протоколом AH или ESP во время обработки пакета в отправной точке защищенного канала. Когда пакет приходит в конечный узел защищенного канала, из его внешнего заголовка ESP или

AH (на рисунке — из заголовка ESP) извлекается значение SPI, и дальнейшая обработка пакета выполняется с учетом всех параметров заданной этим указателем ассоциации.

Таким образом, для распознавания пакетов, относящихся к разным безопасным ассоциациям, используются:

- на узле-отправителе — селектор;
- на узле-получателе — индекс параметров безопасности (SPI).

После дешифрования пакета приемный узел IPSec проверяет его признаки (ставшие теперь доступными) на предмет совпадения с селектором записи SPD для входящего трафика, чтобы убедиться, что ошибки не произошло и выполняемая обработка пакета соответствует политике защиты, заданной администратором.

Использование баз SPD и SAD для защиты трафика позволяет достаточно гибко сочетать механизм безопасных ассоциаций, который предусматривает установление логического соединения, с дейтаграммным характером трафика протокола IP.

VPN на основе шифрования

Более масштабным средством защиты трафика по сравнению с защищенными каналами являются виртуальные частные сети (VPN). В зависимости от используемых технологий безопасности данных сети виртуальные частные сети делятся на два класса:

- *сети VPN на основе разграничения трафика* (см. главу 22);
- *сети VPN на основе шифрования* работают на основе рассмотренной техники защищенных каналов.

Виртуальная частная сеть на основе шифрования может быть определена как совокупность защищенных каналов, созданных предприятием в открытой публичной сети для объединения своих филиалов.

Основной публичной сетью сегодня является Интернет и большинство типов защищенных каналов, стандартизованных сегодня, работают в Интернете «из конца в конец», используя стандартный протокол IP. Защищенный канал может быть образован силами клиента Интернета, и от провайдеров обоих окончаний канала требуется только предоставление стандартного доступа в Интернет. В этом состоит основное преимущество VPN на основе шифрования перед VPN на основе разграничения трафика: первые работают в пределах всего Интернета, в то время как вторые — в пределах сети одного провайдера, поддерживающего MPLS.

Сети VPN на основе шифрования могут быть организованы как силами клиентов, так и силами провайдеров, но последний вариант распространен мало.

Сеть VPN на основе шифрования представляет собой своего рода «сеть в сети», то есть сервис, создающий у пользователей иллюзию существования их частной сети внутри публичной сети. Одним из важнейших свойств такой «частной сети» является защищенность трафика от атак пользователей публичной сети. Сетям VPN доступна не только способность имитации частной сети; они дают пользователю возможность иметь собственное адресное пространство (например, частные IP-адреса, такие как адреса сети 10.0.0.0) и обеспечивать качество обслуживания, близкое к качеству выделенного канала.

Технологии VPN на основе шифрования включают шифрование, аутентификацию и туннелирование.

- Шифрование гарантирует конфиденциальность корпоративных данных при передаче через открытую сеть.
- Аутентификация отвечает за то, чтобы взаимодействующие системы (пользователи) на обоих концах VPN были уверены в идентичности друг друга.
- Туннелирование предоставляет возможность передавать зашифрованные пакеты по открытой публичной сети.

Для повышения уровня защищенности виртуальных частных сетей технологии VPN на основе шифрования можно применять совместно с технологиями VPN на основе разграничения трафика. Технологии VPN на основе разделения трафика иногда критикуют за недостаточный уровень безопасности, считая, что без шифрования трафика персонал поставщика услуг может получить несанкционированный доступ к данным. Действительно, такая вероятность существует, поэтому клиент услуг VPN на основе разграничения трафика, например MPLS VPN, может самостоятельно повысить защищенность своего трафика, прибегнув, скажем, к шифрованию передаваемых данных.

Сейчас наиболее широко используются сети VPN на основе протоколов IPSec и SSL.

Стандарты IPSec обеспечивают высокую степень гибкости, позволяя выбрать нужный режим защиты (с шифрованием или только с обеспечением аутентичности и целостности данных), а также использовать различные алгоритмы аутентификации и шифрования. Режим инкапсуляции IPSec позволяет изолировать адресные пространства получателя (клиента) и поставщика услуг за счет применения двух IP-адресов — внешнего и внутреннего.

Сети VPN на основе IPsec, как правило, строятся по типу CPVPN, то есть как виртуальные частные сети, в которых клиент самостоятельно создает туннели IPSec через IP-сеть поставщика услуг. Конфигурирование сетей VPN на основе IPSec довольно трудоемко, поскольку туннели IPSec двухточечные, то есть при полносвязной топологии их количество пропорционально $N \times (N - 1)$, где N — число соединений. Необходимо учесть еще и непростую задачу поддержания инфраструктуры ключей. Кроме того, протокол IPSec может применяться для создания виртуальных частных сетей, поддерживаемых провайдером (PPVPN), — туннели в них также строятся на базе устройств клиента (CE-based), но эти устройства удаленно конфигурируются и администрируются поставщиком услуг.

На рис. 29.22 показан пример организации виртуальной частной сети на основе шифрования, которая обслуживает сотрудников предприятия, работающих удаленно. В корпоративной сети установлен VPN-шлюз, который объединен с корпоративным файерволом (такое объединение функций не является обязательным, хотя часто встречается). На компьютерах удаленных пользователей установлена программа — PN-клиент. PN-клиент обращается к шлюзу и устанавливает с ним защищенный канал. Шлюз VPN должен обладать высокой производительностью для того, чтобы поддерживать одновременно достаточное количество сеансов с удаленными пользователями. Программное обеспечение шлюза должно также позволять администратору VPN управлять учетными записями удаленных пользователей, а также ключами, применяемыми для аутентификации и шифрования. Учитывая высокий риск ошибки аутентификации удаленного пользователя, эта процедура должна быть максимально надежной: например, двухфакторной аутентификацией с использованием пароля и аппаратного токена доступа.

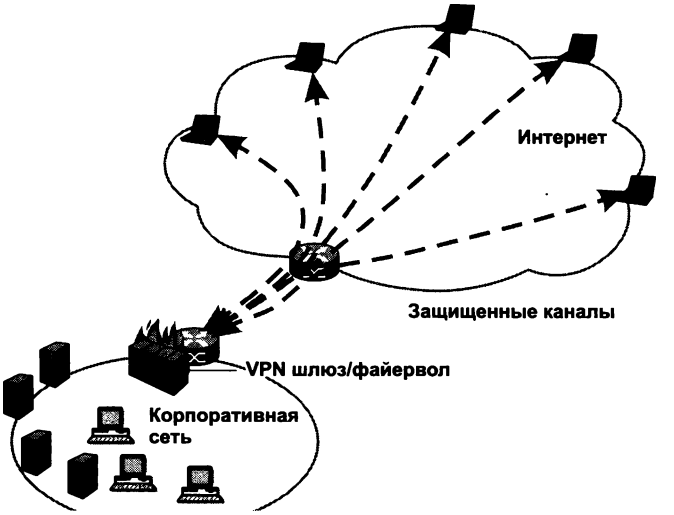


Рис. 29.22. VPN доступа на основе шифрования

В VPN на основе шифрования возможно также использование защищенного канала на основе протокола SSL. Напомним, что этот протокол работает на уровне представления, непосредственно под прикладным уровнем, так что приложения, чтобы создать защищенный канал для своего трафика, должны вызывать его *явным* образом. Наиболее популярным приложением, использующим защищенные каналы SSL, является веб-браузер. Защищенные каналы SSL образуются здесь на основе протокола HTTP, который в этом режиме работы называют протоколом HTTPS. Пользователи Интернета хорошо знают этот режим, так как браузер прибегает к нему во всех случаях, когда необходимо обеспечить конфиденциальность передаваемой информации: при покупках в интернет-магазинах, при интернет-банкинге и т. п.

Служба VPN на основе SSL функционирует на базе веб-портала, развернутого в локальной сети организации. Пользователи такой защищенной службы VPN получают удаленный доступ к ресурсам этой локальной сети, обращаясь к веб-порталу посредством обычного браузера через порт 443 (TCP-порт протокола HTTPS). Отсутствие специального клиентского программного обеспечения, требующего настройки, является значительным преимуществом VPN на основе SSL.

Выводы

Атаки на транспортную инфраструктуру сети разнообразны, и для их классификации могут быть использованы различные критерии: тип атакуемого протокола, цель атаки, способ организации атаки и др. В соответствии со способом проведения атаки делятся на следующие группы:

- *Атаки отказа в обслуживании* реализуются путем отправки на атакуемый компьютер интенсивного потока запросов. Примеры: атака затоплением SYN-пакетами, Ping-затопление, UDP-затопление, атака на опции IP-пакета (отказ в обслуживании маршрутизатора).

- *Атаки, действие которых основано на «отраженном» потоке.* Примеры: ICMP-атака Smurf, TCP-атака затоплением ACK-пакетами, ICMP/UDP-затопление, UDP/echo/chargen-затопление, отраженная атака затоплением SYN-пакетами.

Как прямые, так и отраженные атаки отказа в обслуживании могут быть усилены привлечением к атаке сети ботов, в таком случае говорят о *распределенной атаке*. Примеры: DDoS-атака затоплением SYN-пакетами, DDoS-атака затоплением ACK-пакетами.

Большая группа атак строится на использовании уязвимостей протокола, связанных с *недостаточным контролем параметров*. Примером такой атаки является атака «Пинг смерти», которая отправляет на атакуемый компьютер эхо-запрос с длиной IP-пакета, превышающей максимально возможный для пакета. Аналогично работает атака на IP-фрагментацию.

Другими способами организации атак является вмешательство в логику протокола (сброс TCP-соединения), нарушение целостности передаваемых данных (подделка TCP-сегмента, повторное использование TCP-сегментов).

К основным средствам борьбы с атаками на транспортные протоколы относятся фильтрация, мониторинг и анализ трафика, а также совершенствование коммуникационных протоколов.

Для преодоления защитных фильтров, а также для того, чтобы избавиться от вредящих ему ответных пакетов атакуемого сервера, злоумышленник может прибегать к спуфингу — размещению в поле адреса отправителя поддельного адреса.

Важным инструментом злоумышленника является сетевая разведка — предварительный сбор данных об атакуемой сети: адресах, портах, типах и версиях ОС и приложений.

Среди всех возможных атак на транспортную инфраструктуру наибольшую опасность представляют атаки на глобальную службу имен DNS и атаки на протокол BGP — протокол обмена маршрутной информацией между автономными системами Интернета.

Основной вид атак на DNS-серверы — это атаки отказа в обслуживании. Применяются также атаки отравления кэша DNS, направленные на замену корректной записи с отображением имени в кэше некоторого DNS-сервера поддельной записью, которая направляет DNS-клиента на ложный узел. Аналогичную цель преследует DNS-спуфинг.

Уязвимостью протокола BGP является незащищенность его маршрутных объявлений от подделки. Актуальным средством защиты BGP являются фильтрация маршрутных объявлений и использование цифровых сертификатов.

Технология защищенного канала обеспечивает защиту трафика между двумя точками в открытой транспортной сети, для чего решает следующие задачи: взаимная аутентификация абонентов при установлении соединения, которая может быть выполнена, например, путем обмена паролями; защита передаваемых по каналу сообщений от несанкционированного доступа, например, путем шифрования; подтверждение целостности поступающих по каналу сообщений, например, путем передачи одновременно с сообщением его дайджеста.

Контрольные вопросы

1. В чем состоит главная уязвимость протокола IP? Варианты ответов:
   а) заголовок IP-пакета переносит адреса источника и назначения в незашифрованном виде;

б) он использует широковещательные адреса назначения;

в) он позволяет каждому узлу Интернета взаимодействовать с каждым без предварительного установления соединения;

г) он поддерживает фрагментацию пакетов.

2. С помощью протокола ICMP злоумышленник может:

а) определить, что некоторый хост находится в работоспособном состоянии;

б) организовать DoS-атаку;

в) перенаправить маршрут;

г) узнать, через какие промежуточные маршрутизаторы проходит маршрут до некоторого конечного узла.

3. С какой целью злоумышленник должен подавить отправку ACK-сегментов на атакуемый сервер в ходе атаки SYN Flood? Варианты ответов:

а) чтобы скрыть свой IP-адрес;

б) чтобы открытые соединения оставались незавершенными;

в) чтобы закрыть открытые соединения.

4. Чем атака DNS-спуфинга отличается от атаки отравления DNS-кэша? Варианты ответов:

а) ничем, это разные названия одной и той же атаки;

б) в результате атаки отравления DNS-кэша DNS-сервер терпит крах, в то время как атака DNS-спуфинга к краху сервера не приводит;

в) в атаке DNS-спуфинга ложный ответ передается клиенту, а в атаке отравления DNS-кэша — DNS-серверу;

г) в атаке DNS-спуфинга ложный ответ передается DNS-серверу, а в атаке отравления DNS-кэша — клиенту.

5. Каким образом можно «подделать» маршрутное объявление BGP, которое вы передаете вашему соседу, если ваша автономная система является транзитной для этого маршрута, а вы хотите, чтобы сосед не использовал этот маршрут для передачи трафика:

а) добавить в объявление номер автономной системы вашего соседа;

б) выбросить из последовательности определенную автономную систему;

в) добавить номер своей автономной системы несколько раз;

г) заменить своими адрес сети и номер исходной автономной системы.

6. С какой целью в семействе протоколов IPSec функции обеспечения целостности дублируются в двух протоколах — AH и ESP?

ГЛАВА 30 Безопасность программного кода и сетевых служб

Уязвимости программного кода и вредоносные программы

Программная система, состоящая из десятков тысяч строк кода, всегда имеет уязвимости, которые может использовать злоумышленник. Эти уязвимости могут быть результатом ошибок программистов: в соответствии с исследованием CyLab Университета Карнеги Мэллона в среднем каждые 1000 строк кода содержат 20–30 ошибок, из которых 5 % влияют на безопасность системы, а 1 % открывает возможности для взлома системы.

Уязвимости, связанные с нарушением защиты оперативной памяти

Области оперативной памяти (адресные пространства) отдельных процессов защищены друг от друга. Защита памяти реализуется операционной системой в тесном взаимодействии с аппаратными механизмами процессора. Однако несмотря на это, некорректное использование областей памяти все же может происходить в пределах адресного пространства *отдельного процесса* или в области памяти *ядра* операционной системы. В последнем случае это особенно опасно, так как может вызвать крах всей системы, а не отдельного приложения, как в первом случае.

Переполнение буфера памяти является, наверное, наиболее часто используемой уязвимостью, связанной с нарушением защиты памяти. Мы уже знаем одну такую атаку, которая использует буфер, расположенный в памяти ядра, и приводит к краху всей системы, — это атака «Пинг смерти». Точнее сказать, приводила, так как ошибка в операционных системах, приводящая их к краху при превышении IP-пакетом размера в 65 535 байт, уже давно устранена. Тем не менее механизм, который эксплуатируется этой атакой, очень типичен — он использует отсутствие контроля над вводимой из внешнего мира информацией, в данном случае не контролируется длина помещаемого в буфер пакета.

Переполнение стека является частным случаем переполнения буфера памяти. Этот вид уязвимости часто используется злоумышленниками, чтобы заставить ОС выполнить код злоумышленника. Напомним, что стеком является область памяти с реализацией стратегии записи LIFO (Last In First Out — «последним пришел, первым вышел»). Этот способ записи удобен при многократном вызове функций (подпрограмм), так как он обеспечивает экономичный возврат из вызванной функции в вызывающую.

Типичная структура стека, который растет в сторону меньших адресов (архитектура Intel x86), показана на рис. 30.1. Здесь мы видим стек, содержащий данные одной функции $f_1(A_1, A_2)$. В стек помещены аргументы этой функции, за которыми идет адрес возврата в функцию, ее вызвавшую, — в данном случае это функция `main`, то есть основное тело программы, написанной на языке C. За адресом возврата идет локальная память функции f_1, которая используется для хранения ее локальных переменных и массивов. Указатель стека содержит адрес первого слова свободной области стека.

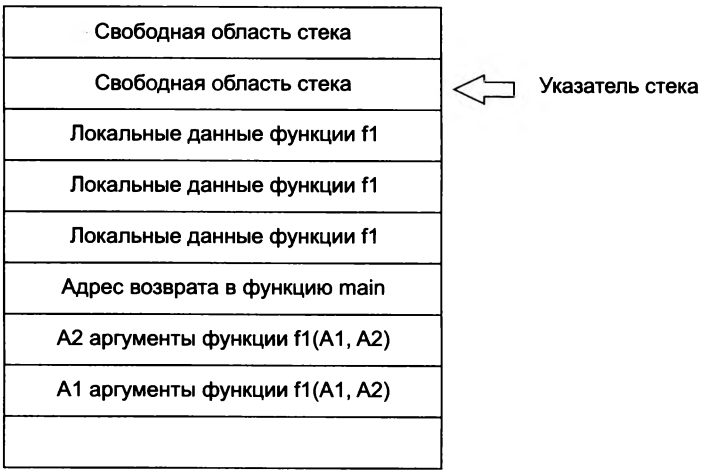


Рис. 30.1. Структура стека до вызова функции f_2 из функции f_1

Если функция f_1 вызывает другую функцию f_2, то ее аргументы, адрес возврата в функцию f_1 и ее локальная память будут размещаться над областью памяти, выделенной в стеке функции f_1 (рис. 30.2).

При завершении функция f_2 должна выполнить специальную инструкцию `RET`, которая вернет управление по адресу возврата в функцию f_1 и очистит стек от данных функции f_2, вернув указатель стека на прежнее место.

Переполнение стека может произойти в случае, когда в область локальной памяти функции помещаются данные, длина которых больше длины этой области. В таком случае эти данные могут наложиться на адрес возврата, в результате после завершения вызванной функции произойдет переход на некоторый адрес, который может быть случайным или специально сформированным злоумышленником. Многие атаки основаны на том, что в область локальных данных стека помещается вредоносный код, которому затем передается управление за счет подмены содержимого поля адреса возврата адресом начала вредоносного кода.

Защитить свою программу от переполнения стека можно разными способами. Одним из них является использование языков программирования, которые автоматически контролируют защиту памяти: например, C# и Java делают это, в то время как C и C++ оставляют это на откуп программисту. Можно не полагаться на компилятор, а включить в код процедуру проверки корректности ввода каждый раз после получения аргумента от пользователя или другой функции.

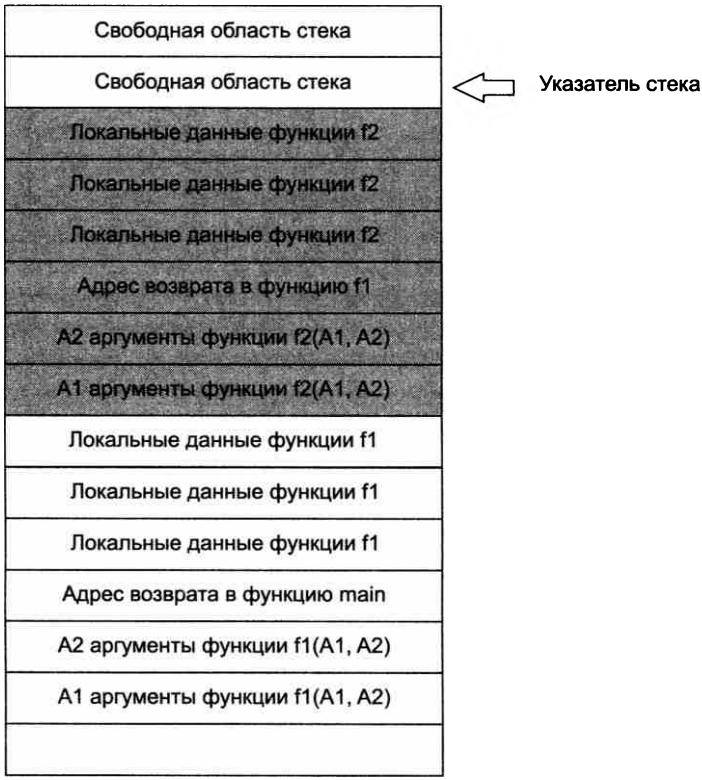


Рис. 30.2. Структура стека после вызова функции f_2 из функции f_1

Некоторые операционные системы помечают область стека как неисполняемую, что предотвращает выполнение вредоносного кода в случае его попадания в стек.

Более сложно злоумышленнику использовать переполнение кучи (heap) — области памяти, динамически выделяемой программе по запросу `malloc`. Переполнение буфера, находящегося в куче, не вызывает краха программы, но разрушает структуры данных, что приводит к неверным результатам выполнения программы.

(S) *Пример использования техники переполнения стека для организации атаки*

(S) *Скрытые коммуникации и скрытые каналы*

Уязвимости контроля вводимых данных

Переполнение буфера является частным случаем уязвимостей, являющихся следствием слабого контроля вводимых данных. В более общем случае — *любая* непредвиденная создателем программы форма вводимых данных может вызвать совершено неожиданные последствия, и этот факт может быть использован злоумышленником. Как любят повторять специалисты по разработке безопасного кода: «Любой ввод данных — это зло!».

Тривиальным примером является веб-форма, в которой пользователю предлагается ввести номер статьи, выбранный из списка, включающего 10 статей. Если разработчик не предвидел, что вместо ожидаемого положительного числа из диапазона от 1 до 10 пользователь может ввести –1, то его приложение может повести себя совсем не так, как он планировал, например выдать конфиденциальный документ вместо публично доступной статьи. Многие системы программирования пытаются исключить такие ситуации за счет того, что заставляют разработчика явно описывать тип вводимых переменных и их возможные значения, но этого не всегда может быть достаточно. Например, тип переменной «текстовая строка» не сможет предохранить приложение от подмены имени пользователя строкой скрипта или строкой HTML-тега, так как все эти переменные являются строками и нужен более детальный анализ содержимого строки, чтобы распознать попытку «подделки» вводимых данных.

Метод борьбы с внедрением вредоносного кода при вводе данных имеется только один — любые данные, которые программа получает от источника, не вызывающего доверия, требуют тщательной проверки перед их использованием. Этот подход аналогичен принципу защиты периметра сети, рассмотренному в главе 27: вся информация, которая приходит извне доверенного периметра, должна тщательно фильтроваться.

На рис. 30.3 показана программа P1, работающая на внутреннем веб-сервере предприятия. Периметр доверия этой программы охватывает программы, работающие на том же сервере, а также программы, работающие на внутреннем SQL-сервере. Поэтому данные, поступающие в программу P1 от программы P2, не подлежат тщательной проверке. Гораздо более строгий подход должен применяться ко вводу данных от пользователя этого предприятия и от программы P3, работающих за пределами периметра доверия.

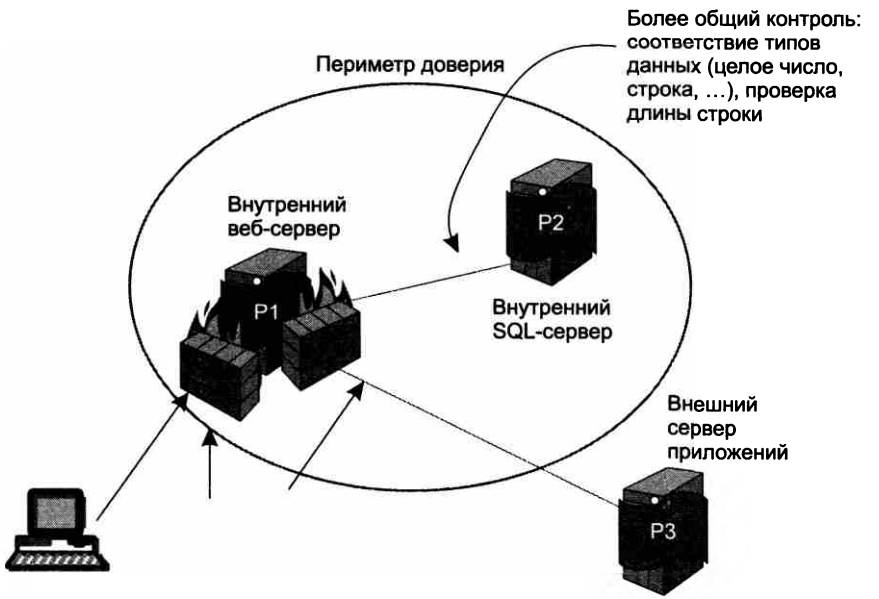


Рис. 30.3. Периметр доверия программы

Фильтрацию вводимых данных может делать как сама программа, так и файервол, работающий на прикладном уровне, а также система обнаружения вторжений (ISD) хоста. Хорошо, когда фильтрацию выполняют все три компонента. Сама программа лучше всего знает специфику вводимых данных и возможные угрозы, в то время как файервол и ISD могут выполнять более общие проверки для определенного типа угроз.

Внедрение в компьютеры вредоносных программ

Многочисленная группа атак связана с внедрением в компьютеры **вредоносных программ** (malware), к числу которых относятся троянские и шпионские программы, черви, вирусы, спам, логические бомбы и некоторые другие типы программ, нацеленные на преодоление системы безопасности. Вредоносный код чаще всего классифицируют по способу проникновения кода в чужой компьютер, а также по целевому назначению.

Самый простой *способ* проникновения — «самодоставка», когда пользователь загружает файлы из непроверенных источников (съемных носителей или веб-сайтов) либо беспечно открывает подозрительный файл, пришедший к нему по электронной почте. Существуют и более сложные представители вредоносных программ, обладающие собственными механизмами «размножения», копии таких программ распространяются по компьютерам сети без участия пользователей.

Ущерб, наносимый вредоносными программами, может выражаться не только в уничтожении, искажении или похищении информации, приведении в нерабочее состояние программного обеспечения, а значит, и компьютера в целом, но и в значительных затратах времени и сил администраторов на обнаружение и распознавание атак, фильтрацию внешних сообщений, тестирование и перезагрузку систем. Вредоносные программы в начале этого десятилетия были одной из основных причин нарушения безопасности компьютерных сетей. Однако, как показала статистика, в последние несколько лет суммарный ущерб, нанесенный вредоносными программами предприятиям, резко снизился. Это связывают, в том числе, с улучшением качества антивирусных средств и ужесточением наказания за такого рода преступления.

На практике злоумышленники часто сочетают в одной и той же программе различные типы угроз. Например, некоторые черви способны маскироваться под троянские программы или подобно вирусам заражать исполняемые файлы на локальном диске, а некоторые вирусы наделены способностями червей самокопироваться на другие компьютеры. Кроме того, вы можете встретить и другую классификацию вредоносных программ, где, скажем, троянские программы и черви рассматриваются как разновидности вирусов.

Троянские программы

Троянские программы, или **трояны** (**trojan**), — это разновидность вредоносных программ, которые наносят ущерб системе, маскируясь под какие-либо полезные приложения.

Троянские программы могут применять в качестве прикрытия знакомые пользователю приложения, с которыми он работал и раньше, до появления в компьютере «троянского коня». При другом подходе в полном соответствии с древней легендой троянская программа при-

нимает вид нового приложения, которое пытается заинтересовать пользователя-жертву какими-то своими якобы полезными функциями.

Однако суть троянской программы в том и в другом случае остается вредительской: она может уничтожать или искажать информацию на диске, передавать данные (например, пароли) с «зараженного» компьютера на удаленный компьютер хакера, приводить в неработоспособное состояние установленное на атакованном компьютере программное обеспечение, участвовать в проведении DoS-атак на другие удаленные компьютеры. Так, одна из известных троянских программ AIDS TROJAN DISK7, разосланная нескольким тысячам исследовательских организаций на дискете, при запуске перемешивала символы в именах всех файлов и заполняла все свободное пространство жесткого диска. После этого программа от имени злоумышленника предлагала помощь в восстановлении диска, требуя взамен вознаграждение для автора этой программы. (Злоумышленники могут также шантажировать пользователя, зашифровывая его данные.) Кстати, описанное компьютерное преступление завершилось поимкой хакера-шантажиста.

Троянские программы могут быть отнесены к самому простому по реализации виду вредоносных программ.

Сетевые черви

Сетевые черви (worm) — это программы, способные к самостоятельному распространению своих копий среди узлов в пределах локальной сети, а также по глобальным связям, перемещаясь от одного компьютера к другому без всякого участия в этом процессе пользователей сети.

Поскольку большинство сетевых червей передаются в виде файлов, основным механизмом их распространения являются сетевые службы, основанные на файловом обмене. Так, червь может рассылать свои копии по сети в виде вложений в сообщения электронной почты или путем размещения ссылок на зараженный файл на каком-либо веб-сайте. Однако существуют и другие разновидности червей, которые для своей экспансии используют более сложные приемы, например связанные с ошибками («дырами») в программном обеспечении.

Главная цель и результат деятельности червя состоит в том, чтобы передать свою копию на максимально возможное число компьютеров. При этом для поиска компьютеров — новых потенциальных жертв — черви задействуют встроенные в них средства.

Типичная программа-червь не удаляет и не искажает пользовательские и системные файлы, не перехватывает электронную почту пользователей, не портит содержимое баз данных, а наносит вред атакованным компьютерам *потреблением их ресурсов*, например, для рассылки спама или проведения массированной атаки в составе ботнета.

При создании типичного сетевого червя хакер прежде всего определяет перечень сетевых уязвимостей, которые он собирается использовать для проведения атак средствами создаваемого червя. Такими уязвимостями могут быть как известные, но не исправленные на некоторых компьютерах ошибки в программном обеспечении, так и пока не известные никому ошибки, которые обнаружил сам хакер. Чем шире перечень уязвимостей и чем более они распространены, тем больше узлов может быть поражено червем.

Червь состоит из двух основных функциональных компонентов: атакующего блока и блока поиска целей.

- *Атакующий блок* состоит из нескольких модулей (векторов атаки), каждый из которых рассчитан на поражение конкретного типа уязвимости. Этот блок открывает «входную дверь» атакуемого хоста и передает через нее свою копию.
- *Блок поиска целей* (локатор) собирает информацию об узлах сети, а затем на основании этой информации определяет, какие из исследованных узлов обладают теми уязвимостями, для которых хакер имеет средства атаки.

Эти два функциональных блока являются обязательными и присутствуют в реализации любой программы-червя. Некоторые черви нагружены их создателями и другими вспомогательными функциями, о которых мы скажем позже.

Упрощенно жизненный цикл червя может быть описан рекурсивной процедурой, состоящей из циклического запуска локатора и атакующего блока на каждом из последующих заражаемых компьютеров (рис. 30.4).

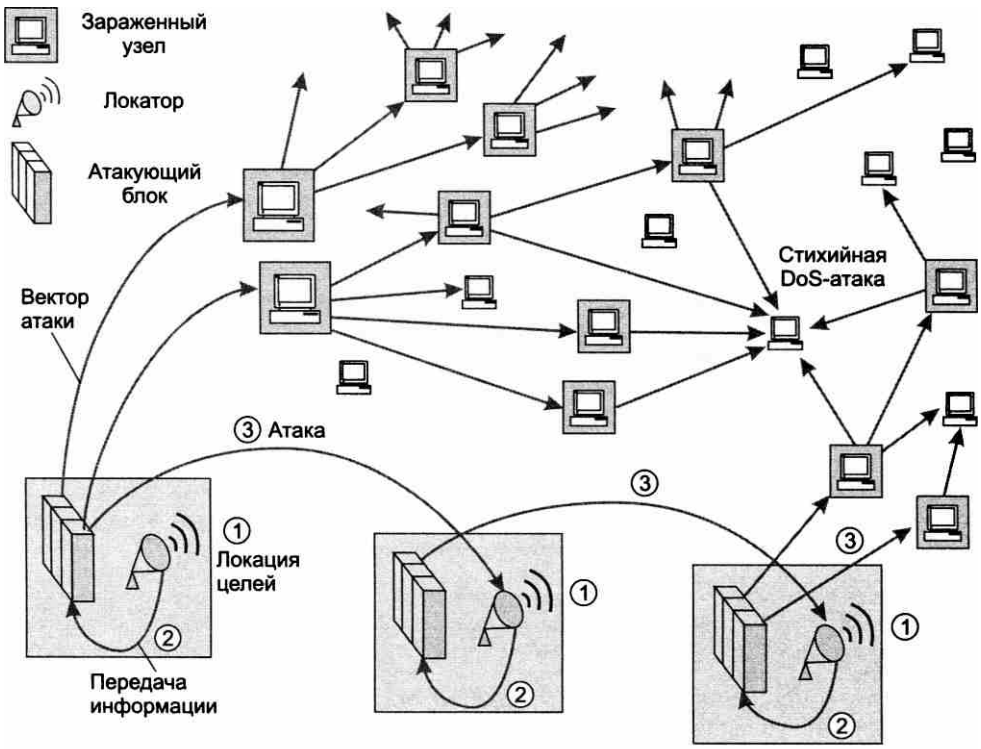


Рис. 30.4. Экспансия червя в сети

В начале каждого нового цикла червь, базирующийся на захваченном в результате предыдущей атаки компьютере, запускает локатор для поиска и формирования списка узлов-целей, пригодных для проведения каждой из специфических атак, а затем, используя средства атакующего блока, пытается эксплуатировать уязвимости узлов из этого списка.

В результате успешной атаки червь копирует все свои программы на «новую территорию» и активирует локатор. После этого начинается новый цикл. На рисунке показано, как червь лавинообразно распространяется по сети. Заражение тысяч компьютеров может занять всего несколько минут. Некоторые виды червей не нападают на уже зараженные и/или подвергающиеся атаке в данный момент узлы. Если же такая проверка не предусмотрена в алгоритме работы червя, то в сети случайным образом могут возникать очаги стихийных DoS-атак.

Локатор идентифицирует цели по адресам электронной почты, IP-адресам, характеристикам установленных на хостах операционных систем, номерам портов, типам и версиям приложений.

Для сбора информации локатор может предпринимать действия, связанные как с поиском интересующих данных на захваченном им в данный момент хосте, так и зондированием сетевого окружения. Простейший способ получить данные локально — прочитать файл, содержащий адресную книгу клиента электронной почты[1]. Помимо почтовых адресов, локатор может найти на узле базирования другие источники информации, такие как таблицы конфигурационных параметров сетевых интерфейсов, ARP-таблицы и таблицы маршрутизации. Зная IP-адреса хоста базирования и шлюзов, локатор достаточно просто может определить IP-адреса других узлов этой сети. Для идентификации узлов локатор может также использовать ICMP-сообщения или запросы ping, указывая в качестве адресов назначения все возможные IP-адреса. Для определения того, какие приложения работают на том или ином хосте, локатор сканирует различные хорошо известные номера TCP- и UDP-портов. Определив тип приложения, локатор пытается получить более детальные характеристики этого приложения.

Например, пусть некая программа-червь имеет в своем арсенале средства для атаки на некоторые версии веб-сервера Apache. Для поиска потенциальных жертв локатор этого червя зондирует узлы сети, посылая умышленно ошибочные запросы к веб-серверу:

```
GET / HTTP/1.1\r\n\r\n
```

Узел, на котором установлен сервер Apache, отвечает на такой запрос так, как и рассчитывал разработчик червя, то есть сообщением об ошибке: например, это может быть сообщение такого вида:

```
HTTP/1.1 400 Bad Request
Date: Mon, 23 Feb 2004 23:43:42 GMT
Server: Apache/1.3.19 (UNIX) (Red-Hat/Linux) mod_ssl/2.8.1
OpenSSL/0.9.6 DAV/1.0.2 PHP/4.0.4pl1 mod_perl/1.24_01
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-1
```

Из этого ответа локатор узнает о том, что на узле установлен веб-сервер Apache версии 1.3.19. Для червя этой информации может быть достаточно, чтобы внести данный узел в число целей.

Собрав данные об узлах сети, локатор анализирует их подобно тому, как это делает хакер при сетевой разведке. Для атаки выбираются узлы, удовлетворяющие условиям, которые

---

[1] Для коллекционирования почтовых адресов локатор может прибегать и к более интеллектуальным методам, которые используют в своей работе спамеры (о спаме см. далее).

говорят о том, что данный узел, возможно, обладает уязвимостями нужного типа (для них в атакующем блоке есть средства нападения). Понятно, что при таком «предположительном» способе отбора целей не всякая предпринятая атака обязательно приводит к успеху. Неудача рассматривается атакующим блоком червя как штатная ситуация, он просто сворачивает все свои действия, направленные на не поддавшийся атаке узел, и переходит к атаке на следующую цель из списка, подготовленного локатором.

Для передачи своей копии на удаленный узел атакующий блок червя часто использует рассмотренную ранее уязвимость *переполнения буфера*.

Помимо локатора и атакующего блока червь может включать некоторые дополнительные функциональные компоненты.

- *Блок удаленного управления и коммуникаций* служит для передачи сетевым червям команд от их создателя, а также для взаимодействия червей между собой. Такая возможность позволяет хакеру координировать работу червей для организации распределенных атак отказа в обслуживании. Сетевые черви могут быть также использованы для организации параллельных вычислений при решении таких требующих большого объема вычислений задач, как, например, подбор секретного ключа шифрования или пароля.
- *Блок управления жизненным циклом* может ограничивать работу червя определенным периодом времени.
- *Блок фиксации событий* используется автором червя для оценки эффективности атаки, для реализации различных стратегий заражения сети или для оповещения других пользователей о повреждениях, нанесенных их компьютерам. Результатом работы данного блока может быть, например, список IP-адресов успешно атакованных машин, посланный хакеру в виде файла или сообщения электронной почты.

Вирусы

Вирус (virus) — это вредоносный программный фрагмент, который может внедряться в другие файлы.

Стремление злоумышленника сделать код вируса как можно более коротким часто ограничивает логику работы вируса очень простыми решениями, которые, однако, иногда приводят к весьма разрушительным последствиям. Так, например, один из реально существовавших вирусов, состоящий всего из 15 (!) байтов, записывал свою копию поверх других файлов в начало каждого сектора диска, в результате система очень быстро терпела крах. Некоторым утешением в таком и подобных ему случаях является то, что одновременно с падением компьютера прекращает свое существование и вирус.

Вирус может внедрять свои фрагменты в разные типы файлов, в том числе в файлы исполняемых программ (рис. 30.5). При этом возможны самые разные варианты: замещение кода, когда размер инфицированного файла не меняется, вставка вирусного кода целиком в начало или в конец исходной программы, замена фрагментов программного кода фрагментами вируса с перестановкой замещенных фрагментов и без перестановки и т. д. и т. п. Более того, код вируса может быть зашифрован, чтобы затруднить его обнаружение антивирусными программами.

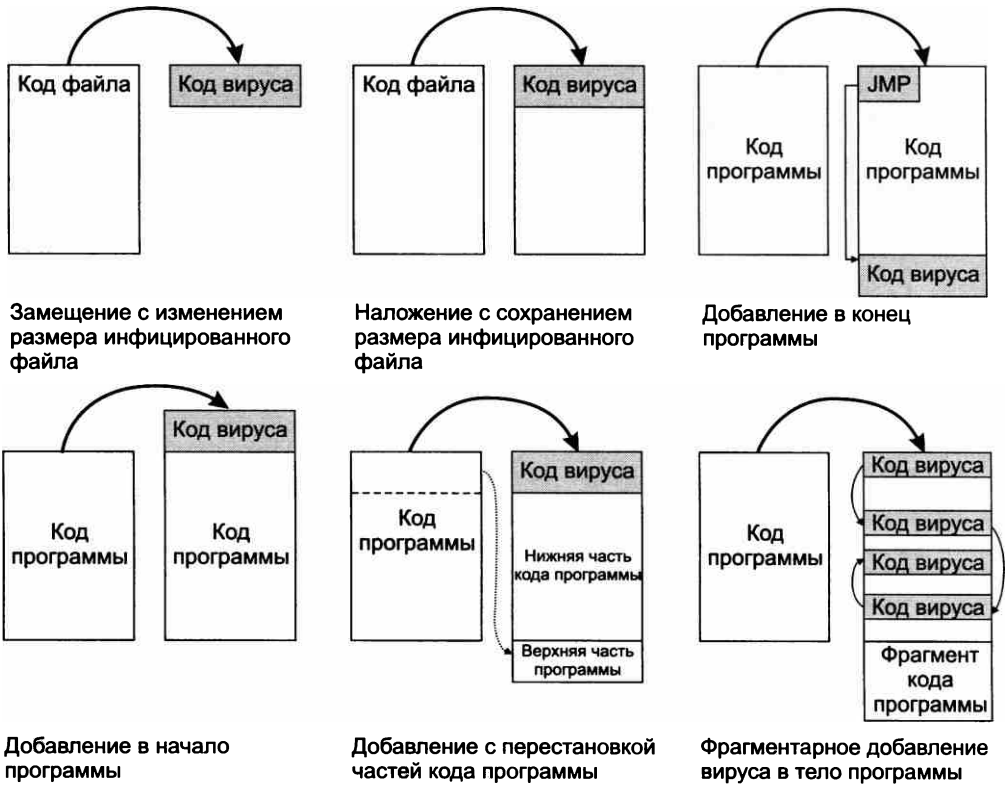


Рис. 30.5. Различные варианты расположения кода вируса в зараженных файлах

В отличие от червей вирусы (так же, как и троянские программы) не содержат в себе встроенного механизма активного распространения по сети, они способны размножаться *своими силами* только в пределах одного компьютера.

Как правило, передача копии вируса на другой компьютер происходит с участием пользователя. Например, пользователь может записать свой файл, зараженный вирусом, на сетевой файловый сервер, откуда тот может быть скопирован всеми пользователями, имеющими доступ к данному серверу. Пользователь может также передать другому пользователю съемный носитель с зараженным файлом или послать такой файл по электронной почте. То есть именно пользователь является главным звеном в цепочке распространения вируса за пределы своего компьютера. Тяжесть последствий вирусного заражения зависит от того, какие вредоносные действия были запрограммированы в вирусе злоумышленником. Это могут быть мелкие, но раздражающие неудобства (замедление работы компьютера, уменьшение размеров доступной памяти, трата рабочего времени на переустановку приложений) или серьезные нарушения безопасности, такие как утечка конфиденциальных данных, разрушение системного программного обеспечения, частичная или полная потеря работоспособности компьютерной сети.

Программные закладки

Программная закладка — это встроенный в программное обеспечение объект, который при определенных условиях (входных данных) инициирует выполнение не описанных в документации функций, позволяющих осуществлять несанкционированные воздействия на информацию. Функции, описание которых отсутствует в документации, называют **недекларированными возможностями**.

Обычно понятие «программная закладка» несет отрицательный смысл, на что указывает прилагательное «несанкционированные» в приведенном определении. То есть подразумевается, что закладка наносит какой-то ущерб системе, на которой она установлена, являясь вредоносным кодом, замаскированным в глубине полезного программного продукта.

Программные закладки могут выполнять различную вредоносную работу, в частности:

- шпионить за действиями пользователя и передавать эту информацию на определенный сервер — это так называемые **шпионские программы** (spyware);
- получать доступ к конфиденциальной информации;
- искажать и разрушать данные.

В то же время недекларированные возможности программы не обязательно являются вредоносными, они могут быть просто дополнительными функциями, которые разработчик программы решил включить для отладки, но не стал их описывать для рядовых пользователей. Это могут быть и просто забытые функции, особенно если речь идет о большой программной системе, в разработке которой участвовали десятки программистов.

Существует также класс недекларированных возможностей программы, внедряемых в нее для развлечения пользователя (и самих программистов тоже) — это так называемые «пасхальные яйца» (Easter Eggs). «Пасхальное яйцо» прерывает нормальную работу пользователя, который, возможно, устал рассматривать ячейки таблицы своего документа, и радостно приветствует его интересной картинкой или сообщением, а то и приглашением поиграть в игру. Авторы наблюдали однажды такое «пасхальное яйцо» в заставке экрана «Трубы» (3D Pipes) ОС Microsoft Windows на экране на несколько минут среди труб появился очень симпатичный чайник. Появился, исчез, и больше мы его никогда не видели, хотя «Трубы» долго работали на наших компьютерах.

Антивирусные программы

Антивирусные программы давно стали необходимым атрибутом жизни любого пользователя. На домашних компьютерах работают индивидуальные пакеты антивирусной защиты, на предприятиях — корпоративные пакеты, состоящие из клиентской программы и сервера, рассылающего обновления. Антивирусные программы используют различные методы для обнаружения вредоносных кодов в файлах, сообщениях электронной почты или HTML-страницах.

Метод сигнатур

Вирус (будем так обобщенно называть далее любой вредоносный код) определенного типа имеет характерную последовательность программных кодов, которая его с какой-то степе-

нью вероятности идентифицирует. Эта последовательность кодов называется **сигнатурой** (подписью) вируса. Для того чтобы обнаружить вирус, антивирусная программа должна иметь библиотеку сигнатур. Постоянное обновление этой библиотеки является одной из самых главных проблем любой компании, выпускающей антивирусное программное обеспечение. Злоумышленники постоянно изобретают новые вирусы, поэтому разработчики антивирусных программ стараются не намного отставать от злоумышленников, своевременно пополняя библиотеку сигнатур. Сервер корпоративной антивирусной системы периодически рассылает обновленные версии такой библиотеки своим клиентам.

Метод сигнатур является основным методом обнаружения вирусов, но он обладает принципиальным недостатком — *неспособностью обнаружить новый тип вируса*.

Кроме того, разработчики вирусов прибегают к *маскировке* сигнатур, что приводит к нераспознаванию вируса. Для этого, например, злоумышленник может использовать *полиморфический код*, когда код изменяет сам себя во время выполнения, что, естественно, приводит к тому, что у него нет постоянной сигнатуры.

Эвристические методы

Эта группа методов строится на более «интеллектуальных» приемах, нежели на простом сравнении инспектируемого кода с большим количеством заранее отобранных сигнатур. Такого рода методы пытаются выявить вирус на основе структуры его кода или его поведения, не имея точной сигнатуры кода, но используя некоторые обобщенные признаки подозрительной структуры кода (*статический анализ*) или подозрительного поведения (*динамический анализ*).

Для безопасного анализа поведения анализируемой программы она помещается в изолированную виртуальную среду, например в среду отдельной виртуальной машины или же созданной программной «песочницы»[1], ограждающей систему от опасных действий программы. В этом случае действия вируса не могут причинить вред основной операционной среде компьютера.

Помещение анализируемой программы в специальную защищенную среду является затратным как по ресурсам, так и по времени. Существует более эффективный, хотя и более рискованный подход, когда анализируемой программе разрешают пробное выполнение в рабочей среде, но при этом антивирусное программное обеспечение следит за всеми ее действиями и в случае необходимости блокирует их, не давая нанести ущерб рабочей среде.

При обнаружении вируса антивирусная программа помещает зараженную программу в карантин и уведомляет об этом пользователя, который принимает решение об удалении зараженной программы или же, если это возможно, удалении из нее вируса.

ПРИМЕЧАНИЕ

Антивирусные программы работают в пространстве ядра, поэтому сами могут причинить ущерб операционной системе из-за своих ошибок. Зафиксированы случаи, когда под видом антивирусной программы пользователям предлагалось вредоносное программное обеспечение.

[1] Песочница (sandbox) представляет собой механизм жесткого контроля набора ресурсов (оперативной памяти, места на диске и др.) и системных сервисов, доступных подозрительной программе.

Ботнет

Бот — это программа, которая выполняет некоторые автоматические (часто интеллектуальные) действия по командам удаленного центра управления.

Бот является *программным роботом*, который может реагировать на возникающую ситуацию и полученные извне команды некоторыми действиями — протоколированием сообщений (полезный бот ведет архив чатов), отправкой сообщений, например поддержанием «разговора» с удаленным собеседником или же участием в DDoS-атаке на какой-то сайт или сеть. Бот может, например, распознавать определенный, заданный ему «хозяином» контекст в дискуссии пользователей социальных сетей Интернета (Livejournal, Facebook) и стать ее участником, выдавая те или иные сообщения. Бот обычно находится в следящем режиме, анализируя сообщения и ожидая команды из центра управления или возникновения заранее определенной ситуации. Есть много похожего у бота и механико-электронного робота — оба запрограммированы на восприятие информации из окружающего мира, анализ ее на предмет обнаружения определенной ситуации и выполнение заранее запрограммированных действий, только набор действий бота ограничен отправкой сообщений либо в сеть, либо операционной системе, в среде которой он находится.

Боты проникают в удаленные компьютеры нелегально, как вирусы, черви или троянские кони. Пользователь может не знать, что его компьютер заражен ботом, потому что компьютеру этого пользователя бот не причиняет вреда, его цели находятся где-то в Интернете. Обычно злоумышленник заражает кодом бота несколько компьютеров, используя различные известные уязвимости ОС и приложений, а затем уже код бота, подобно сетевому червю, пытается заразить как можно больше машин. Зараженная ботом машина обычно называется **зомби**. Группа согласованно работающих ботов называется **ботнетом** (botnet), или **сетью ботов**. Боты часто управляются централизованно, из одного или нескольких центров, являющихся *серверами сети ботов*. Возможны и более сложные зависимости между ботами одной сети с иерархическими или одноранговыми схемами взаимодействия.

Для управления ботами центр управления использует различные протоколы, одним из наиболее распространенных является протокол **IRC** (Internet Relay Chat), позволяющий передавать мгновенные сообщения (чат).

Так как «хозяин» ботнета точно не знает, какие именно машины оказались зараженными кодом бота, для распознавания компьютеров-зомби используются методы сетевого сканирования, например сканирование портов, если код бота слушает определенный порт TCP.

Жертвы «зомбирования» могут составить внушительную армию, способную претворить в жизнь мощную DDoS-атаку, распространить огромное количество спама или осуществить массовый сбор персональных данных. Заметим, что ботнет может работать и как наемная армия — ее «командир» может предоставлять услуги своей сети третьим лицам.

Одним из инцидентов, связанных с пресечением вредоносной деятельности ботнета, была операция, проведенная компанией Microsoft совместно с ФБР в июне 2013 года по разрушению центров управления ботнетами, зараженными *вирусом Citadel*. Этот вирус фиксирует нажатия клавиш на компьютере и передает информацию в свой центр управления. В результате проведенной операции было выявлено и разрушено 1462 центра управления, каждый из которых контролировал свой ботнет. Сообщалось, что эти сети причинили ущерб около 5 миллионам пользователей на сумму свыше полумиллиарда долларов.

Безопасность веб-сервиса

Веб-браузер с его графическим интерфейсом является основным средством доступа пользователя к большинству сервисов Интернета: сайтам новостей, разнообразным справочникам, библиотекам, интернет-магазинам, онлайновым банкам, социальным сетям, таким как Facebook, Twitter, LiveJournal, ВКонтакте, облачным хранилищам информации и приложениям. Даже такие консервативные устройства, как сетевые маршрутизаторы и коммутаторы, стали поддерживать административный доступ посредством веб-интерфейса[1]. Поэтому справедливым будет сказать, что основная часть информации поступает в клиентский компьютер через веб-браузер, а как мы уже ранее отмечали, именно вводимые данные представляют собой главную угрозу для программного обеспечения компьютера. Через веб-браузер попадают в ваш компьютер большинство вредоносных кодов, таких как вирусы, черви и троянские программы, а также назойливые программки, размещающие рекламные объявления на просматриваемой странице без вашего согласия.

Безопасность веб-браузера

Особенностью защиты веб-службы является то, что такая защита требует решения двух достаточно независимых задач:

- обеспечение безопасности программных и аппаратных ресурсов компьютера, на котором эта служба выполняется;
- обеспечение приватности того лица, которое этой службой пользуется.

Приватность некоторого лица — это требование неприкосновенности частной жизни этого лица, включающая запрет на сбор, хранение, использование и распространение информации о его частной жизни без его согласия.

За время, прошедшее с появления первого браузера, разработчики браузеров накопили большой опыт в борьбе со злоумышленниками, и меню *приватности* и *безопасности* современного браузера включает много опций. Рассмотрим, что же стоит за этими опциями, какие риски они стараются снизить и за счет каких средств.

Приватность и куки

Популярность Интернета негативно повлияла на приватность его пользователей. Потенциально все действия пользователя в Интернете — посещенные сайты, просмотренные страницы, запросы поиска — могут быть зафиксированы и проанализированы, и антитеррористические службы, а также службы маркетинга торговых предприятий активно этим занимаются.

Веб-серверы ведут журналы посещений своих сайтов с запоминанием IP-адресов клиентов и предоставляют эти данные владельцам сайтов в удобной форме. Однако анонимность

[1] Это в основном относится к домашним маршрутизаторам, которые рассчитаны на администратора-неспециалиста, которому веб-интерфейс представляется гораздо более удобным, чем командная строка.

в этих журналах до какой-то степени сохраняется, особенно если адрес назначен провайдером динамически.

Браузеры также ведут журналы посещения сайтов и страниц. И если на веб-сервере данные о ваших посещениях, скорее всего, растворились бы в общей статистике, то на вашем компьютере (если он только не разделяется с другими сотрудниками или посетителями кафе или гостиницы) сохраняется история именно ваших посещений и интересов (последнее — в виде запросов к поисковым машинам). Поэтому конфискация компьютера и просмотр журнала истории браузера является теперь одним из первых действий следователя при расследовании дел в отношении подозреваемой личности. В то же время все современные браузеры позволяют пользователю достаточно детально управлять журналом истории посещений, который хранит как адреса посещенных сайтов и страниц, так и кэшированные страницы этих сайтов.

Угрозу приватности несут также куки. **Куки** (cookies — печенье) представляет собой небольшой фрагмент текстовых данных, которым обмениваются веб-сервер и браузер. Куки, относящийся к некоторому сеансу браузера с сервером, содержит информацию о текущем состоянии этого сеанса, аутентификационные данные и персональные настройки клиента, а также уникальный для сервера номер сеанса. В течение всего сеанса куки сохраняется *на стороне браузера*.

При установлении соединения сервер генерирует содержимое куки и передает его браузеру. Веб-браузер, получив текст куки от веб-сервера, сохраняет его в виде файла. В течение всего сеанса пользователя, а возможно, и при всех повторных обращениях данного пользователя к данному сайту браузер передает куки серверу в том же виде, в каком он его получил в последнем ответе сервера. Таким образом достигается эффект запоминания состояния сеанса, причем состояние запоминается на стороне клиента.

Веб-сервер обычно применяет данные куки пользователя для его же (пользователя) удобства: например, интернет-магазины обычно хранят в куки карту покупок пользователя, в них также может храниться история навигации пользователя по страницам сайта. Типичной информацией, помещаемой веб-сервером в куки, является идентификатор сеанса пользователя (SID), на основе которого связываются воедино отдельные запросы пользователя. Даже в случае работы по протоколу HTTP 1.1, который поддерживает длительные TCP-сеансы, эти сеансы могут прерываться из-за временной неактивности пользователя, так что объединение отдельных фрагментов сеанса, чтобы он представлялся пользователю единым, полезно для индивидуального обслуживания пользователя.

Куки бывают *постоянными* — они хранятся в файловой системе ОС и имеют длительные сроки действия и *временными* — их браузер хранит в оперативной памяти и удаляет после своего закрытия.

Куки имеют не только срок, но и *область действия* — она задается доменным именем сайта, который создал куки. Браузер не передает куки сайту с другим доменным именем, но так как доменное имя может быть задано не для конкретного сайта, а для некоторого домена, то есть, например, не для www.cisco.com, а для cisco.com, то куки могут иметь более широкую область действия, чем один сайт.

Так как куки представляют собой текстовые файлы, то угрозы безопасности для пользователя они не несут (за исключением случая, когда в них содержится аутентификационная информация пользователя, — этот случай рассматривается в следующем разделе). Вирусы и другие вредоносные коды с помощью куки не распространяются, так что бытующее мнение, что куки могут заразить компьютер клиента, не соответствует действительности.

В то же время куки могут повредить вашей *приватности*, особенно если в них помещается чувствительная личная информация — данные ваших карт покупок, формы запросов с вашими именем и фамилией, адресом и т. д. Некоторые сайты используют куки третьих сторон, например рекламных компаний. Таким образом с помощью куки ваши предпочтения становятся известны большому количеству сайтов. Самым простым способом защиты своей приватности является полный запрет на прием куки от любых сайтов. Однако при этом вы можете лишиться некоторых удобств, основанных на использовании куки, например тех или иных дополнительных услуг интернет-магазина. Поэтому браузеры оставляют пользователю возможность решать, от каких сайтов он запрещает принимать куки, а от каких разрешает.

Протокол HTTPS

Веб-браузер для взаимодействия с веб-сервером по умолчанию использует протокол HTTP без дополнительных мер по обеспечению основных свойств безопасных коммуникаций, то есть аутентификации сторон, а также конфиденциальности, доступности и целостности данных. Естественно, это создает значительные риски безопасности при работе с сайтами Интернета.

Так, при перехвате злоумышленником незащищенных HTTP-пакетов, циркулирующих между веб-браузером и веб-сервером, вполне возможны атаки вида *человек посередине*. Одной из разновидностей этой атаки является *захват сеанса*, при котором пользователь аутентифицируется на веб-сервере с помощью своего имени и пароля, а затем веб-сервер рассматривает куки, передаваемые в сообщениях браузера, как свидетельство того, что очередной запрос пришел от аутентифицированного пользователя, и продолжает сеанс без повторного запроса пароля. Понятно, что такой способ аутентификации пользователя в случае множественных сеансов протокола HTTP 1.0 или разрыва по какой-то причине длительного сеанса протокола HTTP 1.1 предоставляет злоумышленнику хорошую возможность для захвата сеанса. Для этого ему достаточно перехватить HTTP-запрос, содержащий куки, и затем посылать свои запросы от имени легального пользователя на соответствующий веб-сервер.

Другим вариантом атаки «человек посередине» является атака *повторения*, когда злоумышленник повторяет перехваченные запросы легального пользователя, возможно, несколько модифицируя их. Например, перехватив запросы сеанса пользователя с его банком, злоумышленник может инициировать повторный перевод денег, но теперь уже на свой счет.

В упомянутых примерах злоумышленник использовал уязвимости процесса *аутентификации* пользователя. Очевидно, что прослушивание открытого трафика между браузером и веб-сервером может также нарушить *конфиденциальность* данных и их *целостность*, если злоумышленник по какой-то причине внесет какие-то изменения в данные. Злоумышленник может также нарушить *доступность* данных, просто отбрасывая ответы веб-сайта.

Основным способом обеспечения перечисленных свойств безопасности данных, циркулирующих между веб-браузером и веб-сайтом, является использование **безопасного протокола передачи гипертекста** (Hypertext Transfer Protocol Secure, **HTTPS**) вместо HTTP. Словосочетание «протокол HTTPS» не вполне корректно, поскольку аббревиатура HTTPS подразумевает совместно работающую пару протоколов: HTTP и SSL. Тем не менее название HTTPS прижилось, и пользователь должен его употреблять, когда собирается

инициировать защищенное соединение с веб-сервером, например, вводя адрес https://www.cisco.com. В HTTPS-соединении по умолчанию применяется порт 443 вместо порта 80 в HTTP-соединении.

> В HTTPS-соединении сам протокол HTTP, работающий поверх протокола SSL, остается неизменным. Все атрибуты безопасности коммуникаций — аутентификация, конфиденциальность и целостность — обеспечиваются протоколом защищенного канала SSL (см. главу 29).

Остановимся на некоторых особенностях аутентификации при работе веб-службы. Как вы помните, аутентификация в протоколе SSL основана на цифровых сертификатах. Поэтому при обращении веб-браузера к веб-серверу по протоколу HTTPS каждая из сторон должна иметь подписанный центром сертификации сертификат, достоверность которого можно проверить по цепочке доверия, ведущей к одному из доверенных корневых центров сертификации.

Производители с каждой копией своего браузера поставляют так называемый *встроенный цифровой сертификат*, который может применяться для аутентификации данного браузера. Этот сертификат не аутентифицирует пользователя, работающего с браузером, он служит только для создания защищенного канала при передаче данных между браузером и веб-сервером.

В то же время пользователь может запросить *личный цифровой сертификат* у некоторого центра сертификации и установить его соответствующим образом в своей операционной системе, указав, что он должен применяться для логического входа. В таком случае вход в веб-сервер, требующий аутентификации, может происходить не на основе имени и пароля пользователя, а с помощью этого сертификата, который поставляется браузером серверу по запросу последнего.

Аутентификация сервера при установлении HTTPS-соединения всегда выполняется на основе *цифрового сертификата сервера*, получаемого владельцем сервера. Этот сертификат подтверждает, что данный веб-сервер имеет определенные (одно или несколько) доменные имена.

Браузер обычно уведомляет пользователя о том, что сертификат сервера по какой-то причине является недействительным, оставляя на усмотрение пользователя окончательное решение — отказаться от соединения или все же установить его. Иногда сложный механизм проверки аутентичности сервера работает вхолостую, поскольку пользователи недооценивают угрозы со стороны «невыясненных» веб-серверов и предпочитают действовать на свой страх и риск.

(S) *Проверка действительности сертификата веб-сервера*

Безопасность средств создания динамических страниц

Современные браузеры поддерживают разнообразные средства создания динамических страниц. Все они представляют собой программные коды, полученные извне, и, следовательно, несут риски, связанные с несанкционированным воздействием на клиентский компьютер, начиная с чтения конфиденциальных данных и удаления файлов пользователя и заканчивая разрушением операционной системы.

Из соображений безопасности пользователи должны очень серьезно относиться к любому предложению веб-сайта установить новую *надстройку* или *вставку*, чтобы, например, лучше проигрывать видеоклип определенного формата или же быстрее загружать файлы. Очень может быть, что помимо своей основной функции такая программа будет заниматься еще какой-то побочной деятельностью, наносящей вред вычислительной среде пользователя, например фиксировать нажатия клавиш клавиатуры и передавать их злоумышленнику.

Менее страшны вставки и надстройки, которые изменяют параметры и внешний вид браузера так, чтобы заставить пользователя посещать определенные сайты, обращаться к определенным поисковым системам, ориентированным на рекламу, и пользоваться определенными программами. Этот вид вредоносного программного обеспечения получил название *рекламных вирусов* (Adversary Ware, *AdWare*). Избавиться от паразитов бывает не просто, так как они глубоко встраиваются в операционную систему и часто не удаляются обычными средствами браузера.

Наибольшую опасность для браузера представляют *ActiveX-объекты*, потому что их действия не ограничены никакими рамками, они могут читать, создавать и удалять файлы, а также выполнять любые системные действия. Компания Microsoft снабжает свои ActiveX-объекты цифровой подписью, так же делают и другие производители программного обеспечения, поэтому браузер должен принимать только те ActiveX-объекты, которые подписаны вызывающим доверие разработчиком.

Разработчики *JavaScript-сценариев* и *Java-апплетов*, также применяющихся для создания динамических страниц, встроили в них средства безопасности, что значительно снижает риски, связанные с их использованием.

Браузеры позволяют пользователям управлять процессом создания динамического содержания страницы. Так, пользователь может запретить выполнять ActiveX-объекты или Java-апплеты либо разрешить их выполнение только для доверенных сайтов, список которых он составляет сам.

Безопасность электронной почты

Аналогично защите веб-службы, защита электронной почты также может осуществляться в двух направлениях: обеспечение приватности пользователя (например, конфиденциальности переписки) и обеспечение безопасности ресурсов компьютера (ОС, приложений).

Угрозы приватности почтового сервиса

Пользователи, обменивающиеся сообщениям электронной почты через Интернет, должны принимать во внимание наличие следующих угроз:

- спуфинг имени отправителя — злоумышленник выдает себя за другого пользователя;
- спуфинг почтовых серверов — сервер предъявляет при передаче сообщения ложное имя домена;
- модификация сообщения — искажение или отбрасывание сообщения (то есть нарушение целостности или доступности сервиса);
- утечка информации — чтение сообщения злоумышленником (нарушение конфиденциальности);

- нарушение последовательности сообщений;
- нарушение свойства неотказуемости — отказ отправителя от факта отправки письма, отказ почтового сервера от факта приема письма, отказ получателя от факта получения письма;
- спам — засорение почтовых ящиков пользователей письмами, которые пользователи не просили или же не ожидали получить (обычно спам состоит из рекламных сообщений);
- фишинг — электронное письмо обычно является первым этапом фишинга (напомним, что целью такой атаки является завладение учетными данными пользователя для последующего применения, например для снятия денег со счета, в электронных платежах и т. п.). Такое электронное письмо может выглядеть очень похожим на «настоящее», то есть иметь все атрибуты оформления письма некоторого банка или солидной организации и содержать просьбу обновить свой пароль по приводимой ссылке. Второй этап фишинга выполняет веб-сайт, на который попадает пользователь, перейдя по ссылке;
- нарушение приватности пользователя за счет сбора метаданных почтового сервиса.

Все перечисленные угрозы являются следствием того, что изначально почтовая служба Интернета, основанная на протоколе SMTP, не поддерживала никаких механизмов защиты почтового обмена. Поэтому, например, спуфинг отправителя являлся очень простым делом: почтовый клиент злоумышленника или же его почтовый сервер помещал туда любое имя, требуемое для обмана получателя. Факт такой подмены обнаружить было очень трудно, так как имена пользователей не хранятся в DNS и проверить соответствие IP-адреса имени этим путем невозможно, а аутентификации отправителя по протоколу SMTP предусмотрено не было (только получатель аутентифицировался паролем при получении сообщения).

Аналогично обстояло дело с целостностью и конфиденциальностью переписки, так как текст сообщения в SMTP-пакетах передавался в открытом виде и его легко было прочитать и модифицировать.

Отсутствие аутентификации отправителя приводило к проблемам *неотказуемости* — всегда можно было отказаться от факта отправки письма, сославшись на спуфинг отправителя, мол, это кто-то другой его написал, а меня указал в качестве отправителя.

Квитанция о прочтении письма тоже не является в таких условиях достоверной, так как ее мог сгенерировать злоумышленник, преследуя какую-то свою цель. Отправителю спама также легко было отказаться от авторства рассылки.

К сожалению, применение прошедшего времени в описании такой грустной картины не совсем оправданно — сплошь и рядом электронная почта Интернета используется в своем первозданном виде, несмотря на то что за долгие годы существования этого сервиса разработаны различные стандарты безопасности электронной почты. Велика инерция масштабной распределенной системы интернет-почты — существует огромное количество почтовых серверов, работающих под управлением старых версий программного обеспечения, не поддерживающего новые стандарты, или же под управлением новых версий, в которых новые функции защиты просто не активированы администраторами. Существует очень хороший вариант защиты приватности силами самого пользователя, но он требует от него некоторой дополнительной работы, например получения личного сертификата и установки его в почтовом клиенте.

Далее рассмотрены несколько стандартов безопасности почты Интернета, направленных на снижение рисков, связанных с ее работой.

Аутентификация отправителя

Существует несколько методов аутентификации отправителя:

- ограничение отправителей провайдером услуг;
- аутентификация отправителя провайдером услуг;
- аутентификация отправителя на основе его личного сертификата.

Ограничение отправителей провайдером услуг не является в строгом смысле аутентификацией. Этот способ основан на том, что почтовый сервер провайдера принимает по протоколу SMTP только те письма, которые отправляются клиентами этого провайдера. А принадлежность отправителя к клиентам провайдера проверяется по его IP-адресу — адрес должен принадлежать пулу адресов, которым провайдер владеет и которые он выделяет своим клиентам. Некоторые провайдеры поступают еще строже — они не разрешают своим клиентам пользоваться чужими почтовыми серверами для отправки писем, блокируя соединения на порт 25 от клиентских компьютеров, если они направлены не к почтовому серверу провайдера. То есть провайдер не только блокирует чужих пользователей, но и не разрешает своим пользователям обращаться к почтовым услугам других провайдеров. Тем самым осуществляется взаимная защита провайдеров от чужих пользователей (а также привязка пользователей к провайдеру, что преследует чисто коммерческие цели). Этот метод не гарантирует получателю аутентичности отправителя, но защищает провайдера от спама, отправляемого чужими пользователями.

Аутентификация отправителя провайдером услуг. Расширение протокола SMTP — **SMTP AUTH** — описывает процедуру аутентификации пользователя при отправке сообщения агентом пользователя серверу провайдера почтовых услуг. В соответствии с этим расширением почтовый сервер и агент пользователя в начале SMTP-сеанса договариваются о методе аутентификации. В список возможных методов, в частности, входят: открытый пароль (обычно передается по защищенному каналу SSL), аутентификация на основе слова-вызова и др.

Аутентификация пользователя первым сервером почтовой системы решает многие проблемы: защищает провайдера от спама, позволяет при необходимости решить проблему неотказуемости отправителя. Однако при дальнейшей передаче информация об аутентичности пользователя теряется, поэтому отправитель должен полагаться на добросовестность провайдера, под чьим административным управлением находится почтовый сервер. Даже если провайдер достоин доверия, этот факт не исключает атаки «человек посередине», когда кто-то перехватывает сообщение по пути к почтовому серверу получателя и изменяет имя отправителя.

Аутентификация отправителя на основе его личного сертификата. Этот способ аутентификации работает «из конца в конец», так как сообщение подписывается цифровой подписью отправителя, чей открытый ключ находится в его личном сертификате. Возможность включения цифровой подписи в качестве части сообщения описана в расширении S/MIME, она предусматривает использование различных стандартов цифровой подписи, например **PKCS-7** компании RSA или **PGP (Pretty Good Privacy)**. Аутентификация на основе цифровой подписи отправителя решает несколько задач:

- получатель может проверить аутентичность отправителя и целостность сообщения;
- отправитель не может отказаться от факта отправки письма;

- подпись квитанции о получении/чтении письма делает невозможным отказ получателя от факта получения письма.

Цифровая подпись в расширении S/MIME занимает две части сообщения:

- в первой части описывается используемый стандарт цифровой подписи (протокол) и примененная хеш-функция;
- во второй части, которая является приложением, находится сама цифровая подпись, охватывающая все части сообщения вместе с их заголовками.

В варианте PKCS-7 частью цифровой подписи S/MIME является также цифровой сертификат, выданный одним из сертифицирующих центров, входящих в иерархию PKI. Сертификат удостоверяет принадлежность открытого ключа отправителю, указанному в заголовке почтового сообщения.

Рассмотрим пример электронного сообщения, подписанного по стандарту PKCS-7 почтовым клиентом Microsoft Windows Mail 6.0 и принятого почтовым клиентом Apple Mail 6.6 (сообщение представлено в режиме Raw Source программы Apple Mail 6.6, который показывает все MIME-элементы сообщения).

```
Return-path: <natalia@olifer.co.uk>
Envelope-to: victor@olifer.co.uk
Message-ID: <5ED892093E784C5D9BFD602759D9A7C5@natashaPC>
From: <natalia@olifer.co.uk>
To: "victor" <victor@olifer.co.uk>
Subject: secure email
Date: Sat, 9 Nov 2013 11:05:18 -0000
MIME-Version: 1.0
Content-Type: multipart/signed;
    protocol="application/x-pkcs7-signature";
    micalg=SHA1;
    boundary="----=_NextPart_000_0017_01CEDD3B.940A3930"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Windows Mail 6.0.6002.18197
X-MimeOLE: Produced By Microsoft MimeOLE V6.0.6002.18463
This is a multi-part message in MIME format.
------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: multipart/alternative;
    boundary="----=_NextPart_001_0018_01CEDD3B.940A3930"
------=_NextPart_001_0018_01CEDD3B.940A3930
Content-Type: text/plain;
    charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable
Hi,=20
It is much better to use a secure email correspondence.=20
Enjoy!

------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: application/x-pkcs7-signature;
    name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
    filename="smime.p7s"
MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIITJjCCBDYw
ggMeoAMCAQICAQEwDQYJKoZIhvcNAQEFBQAwbzELMAkGA1UEBhMCU0UxFDASBgNVBAoTC0FkZFRy
```

```
dXN0IEFCMSYwJAYDVQQLEx1BZGRUcnVzdCBFeHRlcm5hbCBUVFAgTmV0d29yazEiMCAGA1UEAxMZ
QWRkVHJ1c3QgRXh0ZXJuYWwgQ0EgUm9vdDAeFw0wMDA1MzAxMDQ4MzhaFw0yMDA1MzAxMDQ4Mzha
```

Здесь мы видим обе части цифровой подписи. Первая часть говорит о том, что это сообщение снабжено цифровой подписью:

```
Content-Type: multipart/signed;
protocol="application/x-pkcs7-signature";
micalg=SHA1;
```

Вторая часть (она отделена пустой строкой) представляет собой собственно цифровую подпись, форматированную алгоритмом base64, который заменил 8-битные коды подписи ASCII-символами.

На клиенте отправителя был установлен личный сертификат, полученный от компании Comodo. Почтовые клиенты автоматически проверяют подлинность сертификата, с помощью которого получена цифровая подпись PKCS-7, поэтому для пользователя этот этап незаметен, в случае положительной проверки он видит только обычное сообщение (помеченное, как правило, особым значком и сообщением «подписано таким-то»). А вот в случае отрицательной проверки, когда сертификат отправителя по какой-то причине оказался недействительным, пользователю-получателю выводится на экран предупреждение, и решение о том, принять сообщение или нет, остается за ним.

Второй вариант цифровой подписи в стандарте S/MIME использует технологию PGP. Система **PGP** заслуживает особого внимания, потому что она была первой системой цифровой подписи и шифрования почтовых сообщений Интернета, использующей технику публичных ключей. Одним из основных отличий подходов, применяемых в PGP и PKCS-7 к получению цифровой подписи, является то, что в PGP принадлежность открытого ключа некоторому отправителю должна быть подтверждена заранее, до получения письма от данного отправителя. Открытые ключи отправителей, с которыми получатель поддерживает защищенную переписку, должны храниться в некотором хранилище, доступном почтовому клиенту получателя. При приходе письма от доверенного отправителя клиентская почтовая программа получателя проверяет подлинность цифровой подписи с помощью открытого ключа отправителя, извлекая его из хранилища.

Для поддержки операции проверки принадлежности открытого ключа некоторому пользователю в PGP вводится понятие **паутины доверия** (Web of Trust). Эта паутина похожа на публичную структуру PKI, так как использует цифровые сертификаты и подразумевает иерархию подписывающих их сущностей, но этими сущностями являются пользователи, которым вы прямо или косвенно (через иерархию доверительных отношений) доверяете. В принципе, пользователь системы PGP волен сам решать, каким образом проверять принадлежность открытого ключа другому пользователю PGP.

Шифрование содержимого письма

Шифрование содержимого письма может происходить как «из конца в конец», так и на отдельных участках маршрута следования письма, например между агентом пользователя и почтовым сервером, принимающим письма от пользователей.

- Шифрование содержимого письма *из конца в конец* предусмотрено спецификацией S/MIME. Она определяет способ шифрования определенной части составного сообщения, причем эта часть шифруется вместе со своим заголовком.

- Шифрование *на отдельных участках* чаще всего осуществляется средствами защищенного канала, создаваемого между двумя непосредственно общающимися сторонами передачи сообщения. Этот канал может быть IPSec- или SSL-каналом в зависимости от предпочтений администраторов сетей, в которых расположены эти стороны. Однако такой способ шифрования не гарантирует конфиденциальности сообщения на всем пути от отправителя до получателя, так как какой-то другой участок пути может не использовать защищенный канал, а протокола общей координации участников распределенной схемы передачи писем пока не существует.

Как и в случае цифровой подписи, для передачи шифрованного сообщения требуются две части — в первой части описывается факт шифрования и его способ, а вторая часть является приложением, в котором находится зашифрованная исходная часть сообщения. Спецификация S/MIME предусматривает использование PKCS-7 и PGP.

Защита метаданных пользователя

Метаданными электронной почты называют характеристики сообщений, которые, не передавая самого содержимого сообщения, определяют адресатов переписки и некоторые другие обстоятельства этого процесса. Точнее, к метаданным электронной почты относят:

- имя отправителя, его почтовый адрес и его IP-адрес;
- имя получателя, его почтовый адрес и его IP-адрес;
- тип данных и их кодировки;
- уникальный идентификатор сообщения и связанных с ним сообщений;
- дату, время и временную зону отправки и получения сообщения;
- форматы заголовков сообщения;
- тему письма;
- статус сообщения;
- запрос на подтверждения получения и открытия письма.

Как видно из описания, сбор метаданных почтового сервиса может дать детальную картину о деятельности некоторого пользователя, даже если он шифрует свои сообщения. При этом собрать их достаточно просто. Во-первых, потому, что метаданные телекоммуникационных сервисов — почты, мобильной связи, веб-сервиса и других — законодательствами большинства стран либо совсем не защищаются, либо защищаются в намного меньшей степени, чем собственно данные сообщений сервиса. То есть в то время как раскрытие содержимого переписки в Интернете требует решения суда, *сбор метаданных не считается атакой* и может проводиться беспрепятственно.

Во-вторых, метаданные именно электронной почты легче привязать к определенному пользователю. Метаданные электронной почты хранятся на компьютерах отправителя и получателя (как и сами сообщения), но, что опасно для приватности пользователей, еще и *в журналах почтовых серверов*, которые передавали эти сообщения. Метаданные пользователей почтового сервиса гораздо легче найти на серверах провайдеров, чем метаданные пользователей веб-сервиса, потому что пользователи почты «привязаны» к определенным почтовым серверам: например, они отправляют почту либо через сервер своего домашнего провайдера, либо через корпоративный сервер, либо через сервер провайдера гостиницы,

вокзала или кафе, где они временно находятся, либо через сервер публичной почты, такой как Gmail. Пользователь получает почту также через вполне определенный сервер, на котором у него имеется учетная запись. Эта ситуация не похожа на веб-сервис, где пользователь может посетить любой сервер Интернета, так что найти следы его посещений путем проверки серверов практически невозможно, даже если пользователь регистрировался на некоторых из них.

О ценности метаданных

Вынужденная отставка генерала Давида Петреуса из-за раскрытия любовной связи с его биографом Полой Бродвел подтверждает важность почтовых метаданных. Петреуса нельзя считать человеком, не осведомленным в вопросах информационной безопасности: после многих лет блестящей военной карьеры, на вершине которой он возглавлял штаб вооруженных сил США, а также был командующим объединенной группировкой войск в Афганистане, Петреус был назначен директором ЦРУ. Тем не менее главный шпион Америки понадеялся на то, что анонимный почтовый аккаунт в Gmail будет вполне безопасен для переписки с Полой, если они не будут отправлять с него писем, а только оставлять черновики писем в локальной папке сервера Gmail. Можно, конечно, сказать, что во всем была виновата Пола, которая начала отправлять с этого аккаунта угрожающие письма Джил Келли, другу семьи Петреусов. Джил заявила об этих письмах ФБР, и это инициировало расследование. Так как в деле было замешано имя директора ЦРУ, расследование было проведено тщательно и выйти на след анонимного пользователя почтового аккаунта помогли почтовые метаданные. Хотя в содержании писем не было никаких «зацепок», позволяющих определить личность автора, агенты ФБР смогли его найти, сопоставив данные логических входов анонима с перемещениями лиц из круга знакомых Петреуса. Выяснилось, что IP-адреса анонима принадлежат нескольким гостиницам, в которых останавливалась Пола точно в те дни, когда аноним входил в свой аккаунт. Этого совпадения оказалось достаточно, чтобы основной подозреваемой стала Пола, ну а дальнейшие доказательства были уже добыты стандартными способами — обысками дома, личного компьютера и допросами.

Ценность метаданных хорошо понимают спецслужбы, недаром одна из программ NSA, о которых рассказал миру Сноуден, называется телефонной и связана с массовым сбором метаданных мобильных пользователей, благо что законы, охраняющие приватность в США, запрещают прослушивание телефонных разговоров, но не запрещают собирать метаданные мобильных клиентов.

Вывод из сказанного прост — защиты почтовых метаданных не существует, шифрование не помогает их скрыть. Возможно, в будущем законодательство в области защиты личных данных будет ужесточено и метаданные станут более защищенными в юридическом отношении.

Спам

Спамом называют рассылку писем большому числу адресатов без их согласия или даже намерения вступить в переписку (по названию постоянно навязываемых посетителям кафе консервов из скетча Монти Пайтон).

Спам является высокодоходным бизнесом, так как считается хорошим рекламным средством, и торговые компании платят за рекламу своих товаров и услуг тем лицам и провайдерам, которые рассылают спам, а затраты на рассылку очень низкие. Правда, компаниям-спамерам приходится предпринимать усилия по составлению списков рассылки, адреса жертв стараются найти различными способами, благо пользователи должны указывать адрес своей почты очень часто — при регистрации в гостинице, при получении доступа к чему-то бесплатному в Интернете, не говоря уже о платных услугах.

Является ли рассылка спама преступлением или нет, определяется законодательством каждой конкретной страны. В начале 2000-х годов во многих странах были приняты акты, определяющие, что является спамом и какие наказания применять за его рассылку. Однако принятие этих актов только незначительно снизило процент спама, в общем потоке электронных писем он по-прежнему очень высок и достигает 80–85 %. Это связано с тем, что определение спама является достаточно безобидным. Например, массовая рассылка не считается спамом, если в письме ясно указана его рекламная цель, а получатель имеет возможность отписаться от рассылки. Кроме того, доказать на практике тот факт, что пользователь не давал согласия на получение письма, сложно.

Со спамом борются провайдеры Интернета. В «Терминах и условиях» их договоров с пользователями обычно есть пункт, запрещающий пользователю рассылать спам. В Интернете существуют так называемые «черные списки» (balcklists) IP-адресов электронной почты и/или доменных имен, с которых рассылался спам и письма с которых рекомендуется блокировать. Наиболее распространенной практикой является ведение черных списков в виде зон системы доменных имен (DNS). Такая практика получила название **DNSBL** (DNS Black Lists). Почтовый сервер провайдера может быть сконфигурирован так, что он автоматически опрашивает какой-либо файл DNS-зоны, содержащий черный список, и блокирует письмо, если адрес или имя его отправителя имеется в списке.

Одним из наиболее часто используемых черных списков спамеров является список некоммерческой компании Spamhouse. В главе 29 рассматривалась мощная DDoS-атака с суммарной интенсивностью трафика в 75 Гбит/с, которой в марте 2013 года был подвергнут веб-сервер этой компании. Атака на Spamhouse была местью одной из компаний, занимающейся рассылкой спама, за включение ее в черные списки. Spamhouse имеет очень мощную распределенную систему DNS-серверов, которые предоставляют по запросу почтовых серверов или клиентов черные списки. Spamhouse ведет несколько таких списков — для спамеров, для хостов, зараженных вирусами, для хостов, не выполняющих аутентификацию при передаче письма на почтовый сервер (это считается нарушением политики безопасности почтового сервиса). Владельцы адресов, попавших в черный список, могут оспорить решение Spamhouse, это нормальная процедура, так как при современном «незащищенном» состоянии почты Интернета ошибки при определении источника спама неизбежны.

Атаки почтовых приложений

Текст почтового сообщения на первый взгляд не может причинить вред компьютеру пользователя. Однако гибкость современной почты позволяет злоумышленникам внедрять в сообщения разнообразную информацию, в том числе исполняемые коды, которые уже не являются столь безобидными.

Проще всего поместить вредоносный код в *приложение* почтового сообщения. Почтовый клиент при открытии пользователем приложения передает его одной из программ клиентского компьютера для обработки. Если приложением является исполняемый файл, например файл с расширением .exe, то почтовый клиент передаст его на выполнение операционной системе. Программа, находящаяся в файле, может содержать вирус или оказаться троянцем. Выполняемая программа может действовать и более грубо, просто стерев пользовательские файлы или файлы операционной системы, если адресат письма вошел в систему как администратор. Расширения выполняемых программ могут быть и другими, например, если это Java-программа или скрипт командного процессора.

Понятно, что открывать приложения, которые представляют собой исполняемую программу, очень опасно, поэтому правилом номер один при работе с почтой является запрет такого действия.

Однако исполняемый код может быть также выполнен в виде макроса или скрипта какого-либо документа, например документа MS Word или Excel. Такое приложение вызывает меньше подозрений (ведь это только текст или таблица), но скрипты документов также могут получить доступ к ресурсам компьютера и причинить ему вред.

Вредоносный код может находиться и в *теле сообщения* (если оно написано на языке HTML) в виде JavaScript-скриптов или, что действительно опасно, — ActiveX-объектов. Поэтому все почтовые сообщения должны проходить обязательную проверку антивирусной программой на наличие вредоносного кода в приложениях и самом сообщении.

Облачные сервисы и их безопасность

Концепция облачных вычислений

До недавнего времени пользователи компьютерной сети твердо знали, на каких компьютерах работают программы, обрабатывающие их данные. Это были либо их собственные компьютеры, на которых пользователи запускали текстовые процессоры или программы подготовки презентаций, либо корпоративный сервер, размещенный в центре данных предприятия.

Новая концепция организации вычислений, получившая название **облачных вычислений** (cloud computing), изменяет привычный мир пользователя, так как в соответствии с ней компьютер, который выполняет программу пользователя, находится где-то в «облаке» вычислительных ресурсов — процессоров, оперативной памяти, дисковых накопителей. Облачные вычисления позволяют разгрузить пользовательский компьютер и перенести вычисления на некоторые удаленные компьютеры, связанные с пользовательским компьютером через сеть. Это облако может принадлежать как коммерческому провайдеру, так и самому предприятию, но его организация скрыта от пользователя. При этом пользователь не заботится о том, на каком именно компьютере он должен запустить свою программу, достаточно ли у компьютера ресурсов для качественного выполнения программы в данный момент, достаточна ли пропускная способность сети для быстрого получения реакции программы на своем мониторе — он просто запускает программу, а затем работает с ней, как если бы работал с ней локально. Результаты также можно хранить в облаке, это надежнее, так как провайдер облачных вычислений заботится об их сохранности.

ПРИМЕЧАНИЕ

Сама по себе интерпретация сети как облака не нова, эту метафору применяли всегда, подчеркивая тот факт, что внутренняя организация сети не важна для конечных пользователей, для них важен тот факт, что сеть может передавать данные между двумя компьютерами. Такое облако можно назвать «транспортным». Облачные сервисы имеют дело с «вычислительным» облаком, то есть с облаком, ориентированным на вычисления.

Облачные вычисления являются весьма новым видом услуг, но, по прогнозам специалистов, они станут значительной вехой в процессе эволюции компьютерных сетей. Можно сказать, что с появлением этих услуг эволюция компьютерных сетей завершила виток, вернувшись к первоначальной модели, когда сеть соединяла терминал пользователя с мэйнфреймом, на котором пользователь запускал свою программу. Это действительно так, но как и всегда, виток эволюции привел к появлению нового качества, так как теперь «тупые» терминалы превратились в мощные компьютеры, перепоручающие облачным серверам выполнение только некоторых, возможно критичных, программ, оставляя возможность работать с остальными программами локально.

Хорошую аналогию, поясняющую значение облачных вычислений для развития общества, дал Николас Карр в своей книге *The Big Switch*[1]. Он сравнил появление облачных вычислений в информационном веке с электрификацией в индустриальном веке. До электрификации страны каждая организация, каждый производитель и каждый домовладелец должны были сами заботиться о выработке энергии — для этого существовали водяные колеса, ветряные мельницы, паровые машины. С распространением электрификации исчезла

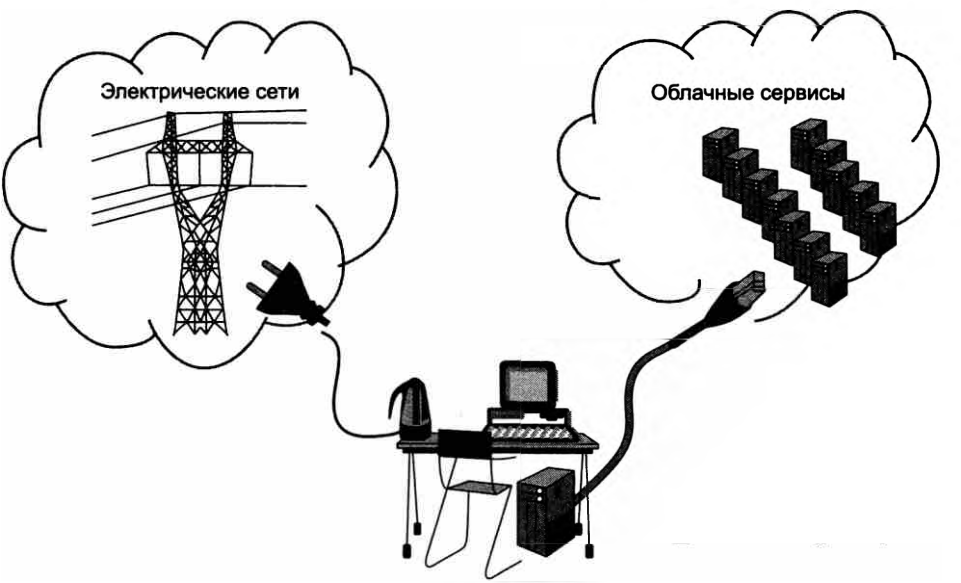


Рис. 30.6. Стратегическое значение облачных вычислений

[1] http://www.amazon.co.uk/The-Big-Switch-Rewiring-Edison/dp/0393333949.

необходимость вырабатывать энергию самостоятельно, достаточно было подключиться к электрической сети. Карр считает, что облачные вычисления открывают новую эру в использовании компьютеров. Сейчас организации обеспечивают себя компьютерными ресурсами самостоятельно. В будущем предполагается, что организация будет просто подключаться к облаку и получать вычислительные ресурсы, которые ей нужны. И если нужно быстро увеличить вычислительную мощность или объем хранимых данных, то облако легко предоставит эти дополнительные ресурсы своему клиенту точно так же, как электрическая сеть легко справляется с дополнительной нагрузкой, когда вы включаете микроволновую печь или электрическую дрель в дополнение к работающему телевизору и настольной лампе (рис. 30.6).

Определение облачных вычислений

Существуют различные определения облачных вычислений. Приведем определение организации NIST, активно участвующей в разработке этого нового вида услуг:

> Облачные вычисления — это модель, предоставляющая удобный доступ по требованию к разделяемому пулу конфигурируемых вычислительных ресурсов, которые могут быть быстро выделены пользователю и отданы обратно в пул с минимальными затратами на управление этим процессом или с минимальным взаимодействием с провайдером услуг.[1]

С этим определением связаны следующие свойства облачных вычислений:

- *Качественно новый уровень разделения ресурсов.* В отличие от прежних моделей вычислений, в которых вычислительные ресурсы принадлежали одному владельцу (ресурсы корпоративной сети, хотя и разделяются между пользователями, принадлежат одной и той же организации-владельцу), облачные вычисления основаны на модели, где вычислительные ресурсы разделяются между многими арендаторами-клиентами и владельцем-провайдером на всех уровнях — уровне сети, хоста и приложений.
- *Высокая степень масштабируемости.* Хотя отдельная организация может владеть сотнями или даже несколькими тысячами серверов, облачная среда позволяет масштабировать вычислительную мощность до десятков тысяч серверов, в разы наращивая пропускную способность каналов доступа и объем хранилища данных. Для обеспечения такой масштабируемости провайдер создает большое количество центров данных, распределенных географически.
- *Эластичность.* Пользователи могут быстро наращивать и уменьшать вычислительные ресурсы по мере необходимости. Таким свойством обычная корпоративная информационная система не обладает.
- *Гибкость оплаты использованных ресурсов.* Пользователи платят только за те ресурсы, которые они действительно задействовали, и за тот период времени, в течение которого ресурсы использовались.
- *Самостоятельное выделение ресурсов.* Пользователи могут управлять выделением необходимых им ресурсов самостоятельно через удобный интерфейс, предоставляемый

[1] The NIST Definition of Cloud Computing, Special Publication 800-145, NIST, August 2011.

провайдером. Скорее всего, эти операции будут выполнять сотрудники ИТ-отдела предприятия, а не рядовые пользователи.

Облачные вычисления появились как результат развития технологий, применяющихся в коммерческих центрах данных при оказании разнообразных услуг хостинга. В конце концов сетевое сообщество осознало, что концепция использования компьютерной сети стала меняться — вместо гибкого соединения жестко заданных пользователем узлов (как правило, клиентского компьютера и сервера) теперь происходит гибкое соединение клиентского компьютера с гибко настраиваемым пулом вычислительных ресурсов. При этом пул ресурсов может быть рассредоточен по различным центрам данных, находящимся в разных городах и, возможно, странах.

Важно понимать, что облачные вычисления не столько новая технология, сколько *комбинация* уже существовавших до этого технологий. Эти технологии-компоненты развивались каждая своими темпами и в разных условиях, изначально они не создавались для работы как единое целое. Однако их смогли «притереть» друг к другу и создать новое качество — облачные среды. Новые достижения в области процессоров, технологий виртуализации, в системах дисковой памяти, широкополосном доступе в Интернет, а также быстрые и недорогие серверы внесли свой вклад в превращение облачных вычислений в очень привлекательную модель.

Основополагающей технологической платформой, на которой базируются облачные вычисления, является **виртуализация**. Термин «виртуализация» относится к абстрагированию компьютерных ресурсов (процессора, памяти, дисковой памяти, сети, стека протоколов и баз данных) от прикладных программ и конечных пользователей облачного сервиса. Технологии виртуализации позволяют многочисленным арендаторам облака видеть ресурсы облака как ресурсы, выделенные только для них. Виртуализация применяется и в центрах данных, принадлежащих предприятию, — она улучшает использование ресурсов и упрощает операционную эффективность ИТ-отдела. Технологии виртуализации применимы к ресурсам разного вида: компьютеру и ОС, хранилищам данных и базам данных, приложениям и сетям (VLAN, MPLS VPN 2-го и 3-го уровня и др.).

Модели сервисов облачных сервисов

По способу реализации облачные среды делятся на публичные, частные и гибридные.

- *Публичные облака* создаются провайдерами услуг и их сервисы предоставляются предприятиям или частным лицам как публичным клиентам.
- *Частные облака* создаются некоторыми предприятиями для использования только сотрудниками этих предприятий, то есть как часть корпоративной сети предприятий.
- *Гибридные облака* — это комбинация публичных и частных облаков, которыми пользуется некоторое предприятие.

Далее рассматриваются модели сервисов, реализуемых в публичных облачных средах.

На сегодня существуют три основные модели сервисов, предоставляемых провайдерами облачных вычислений:

- приложения как сервис;
- платформа как сервис;
- инфраструктура как сервис.

Приложения как сервис (Software-As-a-Service, **SaaS**). Традиционный метод покупки программного обеспечения заключается в загрузке программы в собственный компьютер после оплаты лицензии на применение этой программы (капитальные затраты). Пользователь также может купить контракт на обслуживание, чтобы получать «заплатки» и обновления программы. Пользователь сам заботится о совместимости программы с операционной системой, установке обновлений и удовлетворении условий лицензии.

В модели SaaS пользователь не покупает программное обеспечение, а *арендует* его для работы на условиях подписки или оплаты за использование. То есть вместо капитальных затрат пользователь несет текущие (операционные) расходы. В некоторых случаях при ограниченном применении сервиса (например, программа выполняется не более часа в сутки) сервис может быть бесплатным.

Нужно подчеркнуть, что модель приложения как сервиса отличается от услуг хостинга приложений в двух существенных аспектах. Во-первых, в модели SaaS приложение используется в режиме разделения времени всеми арендаторами (то есть различными организациями), которые подписались на доступ к данному приложению, в то время как приложение хостинга выделяется в единоличное пользование организации-клиенту (хотя и разделяется между сотрудниками этой организации). Во-вторых, программы для хостинга приложений часто пишутся без учета работы через сеть, в то время как программы модели SaaS всегда оптимизированы для сетевого доступа. Примером SaaS-сервисов являются сервисы Google Apps, Microsoft office 365, Apple iWork.

Платформа как сервис (Platform-AS-a-Service, **PaaS**). В этой модели провайдер предоставляет среду для разработчиков программного обеспечения; как правило, в нее входят набор средств разработчика (SDK) и стандарты разработки программ, а также каналы распространения этих программ и механизмы оплаты. Этот сервис направлен на поддержку быстрой разработки приложений с низким уровнем начальных вложений и применением устоявшихся каналов для быстрого нахождения пользователей новых программ. Примером этого типа облачных сервисов является Microsoft Azure.

Инфраструктура как сервис (The Infrastructure-As-a-Service, **IaaS**). В традиционной модели хостинга провайдер предоставляет в распоряжение клиента *выделенную* инфраструктуру для того, чтобы клиент выполнял на ней свои приложения. Модель IaaS тоже обеспечивает клиента необходимой инфраструктурой для выполнения приложений, но эта инфраструктура не является выделенной, она динамически изменяется в зависимости от требований клиента и оплачивается по схеме «оплата за использование». С точки зрения провайдера, такая модель может быть реализована на инфраструктуре, способной гибко реагировать на пики и спады требований каждого клиента. Клиент платит за количество потребленных процессорных циклов, оперативной и дисковой памяти, объем сетевого трафика, но не заботится о физической природе ресурсов — делении диска на разделы, способе увеличения объема памяти, резервировании ресурсов, резервном копировании данных. Провайдер имеет полный контроль над ресурсами инфраструктуры. Клиент, в свою очередь, имеет контроль над тем, какие операционные системы работают на виртуальных машинах инфраструктуры и какие приложения работают под управлением этих ОС. Примером этого типа облачных сервисов является Amazon Web Services.

В зависимости от выбранной модели облачных сервисов провайдер и клиент имеют различные зоны контроля над компонентами облачных вычислений. Эти различия иллюстрирует рис. 30.7. Для каждой модели показано распределение зон управления и ответственности

между провайдером и клиентом для пяти основных компонентов программно-аппаратного комплекса ИС:

- сетевой инфраструктуры (маршрутизаторы, коммутаторы, линии связи);
- хранилища данных;
- серверов;
- виртуальной машины с операционной системой;
- приложений.

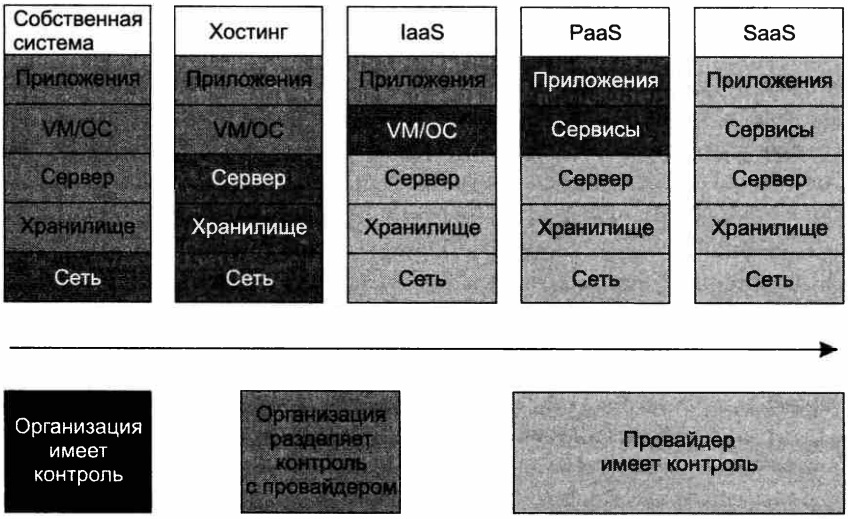


Рис. 30.7. Распределение контроля и ответственности между провайдером и клиентом в разных моделях облачных сервисов

Как видно из рисунка, модель SaaS является крайним случаем, когда провайдер имеет полный контроль над всеми компонентами модели.

При использовании услуг модели IaaS клиент имеет контроль над приложениями, работающими в облаке, а провайдер — над тремя нижними слоями модели. Клиент и провайдер разделяют контроль над виртуальной машиной и операционной системой, работающей в среде этой виртуальной машины.

Модель PaaS занимает промежуточное положение, здесь провайдер и клиент разделяют контроль как над средствами виртуализации программного обеспечения (виртуальной машиной, ОС), так и над самими прикладными программами, поскольку хотя они и разрабатываются специалистами предприятия-клиента, в них работают программные модули провайдера, собранные в единую программу по методике провайдера.

ПРИМЕЧАНИЕ

Все три рассмотренные модели облачного сервиса являются развитием популярных моделей хостинга, таких как хостинг аппаратных серверов, программных веб-серверов и приложений.

Облачные вычисления как источник угрозы

Ограниченная подконтрольность провайдера

Модель облачных вычислений существенно отличается от традиционной модели вычислений, используемой сегодня в корпоративных ИС, и это отличие прежде всего сказывается на обеспечении безопасности ИС. Природа данного отличия довольно проста — вместо того, чтобы строить собственную информационную систему и управлять ею силами сотрудников предприятия, предприятие начинает пользоваться услугами информационной системы, созданной посторонней организацией-провайдером. При этом организация-провайдер владеет всей инфраструктурой ИС и управляет ею, обеспечивая в том числе безопасность данных, принадлежащих предприятию-клиенту. Сотрудники предприятия-клиента используют свои компьютеры только как терминалы доступа к облаку, а все данные предприятия, включая личные данные сотрудников, хранятся и обрабатываются «где-то там, в облаке». Предприятию-клиенту остается только следовать совету Рональда Рейгана: «доверяй, но проверяй».

Такой революционный переворот в модели вычислений не мог не обеспокоить специалистов по безопасности, и многие из них согласны в том, что облачные вычисления — вещь хорошая, но *отсутствие гарантий безопасности* облачных вычислений сводит на нет все преимущества облака.

Для этих опасений, безусловно, есть основания, однако многое зависит от типа организации-клиента и модели облачных вычислений, которую клиент собирается использовать. Понимание моделей облачных вычислений — необходимое условие для правильной оценки рисков предприятия при переходе на новый тип ИС.

Предприятие-клиент облачных сервисов имеет весьма *ограниченный контроль* над механизмами безопасности своих данных, обрабатываемых виртуальными машинами провайдера и хранящимися в виртуальных хранилищах. Особенно это справедливо для услуг модели SaaS, когда защита всех элементов ИС, включая прикладные программы пользователя, осуществляется провайдером. Предприятие-клиент в этом случае участвует лишь в обеспечении безопасности компьютеров своих сотрудников, которые применяются как терминалы облачной среды.

Понятно, что клиент IaaS-сервиса должен самостоятельно заботиться о безопасности своих приложений — следить за тем, чтобы обновления приложений периодически получались и устанавливались, устанавливать и обслуживать антивирусные программы и программы блокировки спама, выполнять все остальные действия в соответствии с политикой безопасности предприятия, которые относятся к приложениям. Обычно в этой модели конфигурирование средств безопасности ОС также является делом клиента.

В модели PaaS контроль над средствами безопасности верхних уровней разделяется между провайдером и клиентом.

В таких условиях клиент должен стараться получить как можно более *полный доступ к средствам аудита провайдера* — к сообщениям и журналам средств безопасности, его виртуального файервола, системы IDS, антивирусных и антиспамовых программ и т. п. Кроме того, полезно также проводить *аудит работы самого провайдера*, привлекая для этого сторонние фирмы, пользующиеся устойчивой репутацией в этой области.

Получение информации от средств защиты провайдера в реальном времени (мониторинг) и доступ к историческим данным этих средств должен быть предусмотрен в *договоре о предоставлении услуг провайдером*. Этот важный документ должен оговаривать все детали взаимоотношений провайдера и клиента, а так как облачные услуги являются новым видом телекоммуникационных услуг, то внимание ко всем его деталям должно быть самое пристальное.

(S) *Соглашение об уровне обслуживания с провайдером облачных сервисов*

Наличие сертификатов на соответствие средств безопасности провайдера популярным программам сертификации также может частично компенсировать отсутствие полного контроля над этими средствами.

Разделение сложной инфраструктуры провайдера

При использовании услуг облачного провайдера вы *разделяете его инфраструктуру* с другими арендаторами, которых вы не знаете. При этом разделение ресурсов провайдера осуществляется не на физическом уровне, а с помощью механизмов виртуализации. Злоумышленник может заключить договор с вашим провайдером и попытаться использовать бреши в механизмах виртуализации для получения несанкционированного доступа к вашим данным. Кроме того, нельзя исключать ошибок персонала провайдера, в результате которых виртуальные барьеры могут быть нарушены.

Инфраструктура облачного провайдера намного *сложнее* инфраструктуры стандартного центра данных, и это представляет собой дополнительную угрозу безопасности облачных сервисов. Кроме таких стандартных элементов виртуализации, как гипервизоры, виртуальные машины, виртуальные маршрутизаторы и файерволы, подобная инфраструктура включает многочисленные дополнительные компоненты управления, такие как средства самостоятельного динамического выделения ресурсов клиентами, измерители потребления ресурсов, средства управления квотами, нагрузкой, мониторинга качества, услуг и т. п. Кроме того, облачные сервисы могут быть реализованы в облачной среде другого провайдера, что еще более усложняет картину.

Обычно администраторы корпоративных ОС и приложений получают доступ к ним через локальную сеть. При использовании облачных сервисов административный *доступ должен выполняться через Интернет*, что несет дополнительные угрозы. Необходимо убедиться, что облачный провайдер поддерживает только хорошо защищенные соединения для предоставления административного доступа своим клиентам.

Угроза конфиденциальности персональным данным

При использовании услуг корпоративной сети персональные данные сотрудников предприятия хранятся в справочной службе предприятия (например, работающей на основе Microsoft Active Directory) и предприятие несет ответственность за их конфиденциальность в соответствии с законами и правовыми актами. В том случае, когда сотрудники пользуются услугами облачного провайдера, их персональные данные (имена и пароли) хранятся в справочной службе провайдера. Так как ответственность за их конфиденциальность в конечном счете все равно несет предприятие, необходимо убедиться, что провайдер надлежащим образом обеспечивает их конфиденциальность как при передаче личных данных через Интернет, так и при хранении их в разделяемой между арендаторами справочной

службе провайдера. Для повышения уровня защиты личных данных можно применять раздельные наборы данных для локальной аутентификации пользователей и их аутентификации у облачного провайдера. Но это довольно громоздкое решение, и для большой организации оно может оказаться неработоспособным.

Другим решением является применение схемы федеративной аутентификации. В этом случае личные данные пользователей хранятся в справочной службе предприятия, а служба аутентификации провайдера взаимодействует со службой аутентификации предприятия через защищенное соединение Интернета.

Мультинациональность облачных услуг, законодательство и политика

В мире облачных вычислений существуют различные варианты реализации сервисов в отношении того:

- где данные физически размещены;
- где они обрабатываются;
- откуда происходит доступ к этим данным.

Зачастую эти три точки находятся в разных странах, в каждой из которых действует свое законодательство. В разных странах также могут быть зарегистрированы предприятие-клиент и облачный провайдер. Пока что законодательные аспекты таких многонациональных услуг определены плохо, эта работа находится в своей начальной стадии, в результате появляется много неясностей во взаимоотношениях провайдеров и клиентов многонациональных облачных услуг, а значит, риски использования этих услуг весьма высоки. Снизить риски, связанные с многонациональными облачными сервисами, можно за счет непосредственного указания в договоре конкретной судебной инстанции определенной страны, которая будет разбирать любые споры между провайдером и клиентом, если они возникнут.

Для того чтобы облачные вычисления могли успешно развиваться как глобальная, не знающая границ услуга, они должны быть отделены от политики. К сожалению, сегодня законы, принимаемые разными правительствами, часто оказывают негативное влияние на развитие глобального облака. Например, одним из результатов принятия Соединенными Штатами Патриотического акта 2001 году стало то, что Канада решила не использовать серверы Интернета, расположенные на территории США, опасаясь за конфиденциальность данных, которые Канада хранит на своих компьютерах. Разоблачения Сноудена в отношении программы PRISM привели Бразилию к решению построить собственный сегмент Интернета с основными сервисами, поддерживаемыми серверами, находящимися на территории Бразилии.

Поэтому выживание облачных вычислений в значительной степени зависит от глобальной политики.

Облачные сервисы как средство повышения сетевой безопасности

Несмотря на то что облачные сервисы принято рассматривать как источник новых угроз безопасности, они могут существенно *улучшить информационную безопасность* предпри-

ятия, особенно если это небольшое предприятие, у которого нет специального подразделения, занимающегося безопасностью. Существует несколько преимуществ облачной модели над традиционными подходами к организации вычислений. Некоторые из них оказываются потенциальными, так как предполагают соответствующую корректную организацию процессов управления безопасностью провайдером услуг, другие являются следствием самой парадигмы облачных вычислений.

Избыточность и резервирование ресурсов

Высокая степень масштабирования ресурсов облачных провайдеров обеспечивает также *высокую доступность* этих ресурсов и, как следствие, данных, обрабатываемых этими ресурсами. Избыточность и резервирование ресурсов являются одними из основных принципов построения облачной среды. Обычно облачный провайдер располагает большим количеством центров данных в разных географических точках, а возможно, и странах, при этом реплики данных одного и того же клиента хранятся в нескольких таких центрах. Это требуется как для повышения производительности за счет приближения данных к пользователям (если это данные публичные, предназначенные для всех пользователей Интернета), так и для обеспечения доступности данных в случае технических отказов или природных катастроф. Кроме того, избыточность ресурсов вызвана необходимостью обеспечивать эластичность сервисов, то есть возможность быстрого увеличения нагрузки по запросу клиента. В корпоративной сети такую высокую доступность ресурсов обеспечить трудно, так как она чаще всего будет экономически неоправданна.

Поглощение DDoS-атак

Распределенная избыточная инфраструктура центров данных облачного провайдера позволяет *эффективно бороться с DDoS-атаками*. Отражение мощной DDoS-атаки является, наверное, наиболее сложной задачей для администратора корпоративной сети, так как фильтрация потока пакетов интенсивностью в десятки, а то и сотни Гбайт/с, направленного на единственную копию сервера через канал связи, требует наличия очень производительного файервола. Такие файерволы, специально созданные для отражения DDoS-атак, существуют. Однако даже если предприятие может себе позволить приобрести и установить столь дорогостоящее устройство, остается проблема узкого места, которое представляет собой единственная копия атакуемого сервера (скорее всего, это веб-сервер, возможности которого публиковать открытые данные злоумышленник пытается подавить) и единственный канал связи сервера с Интернетом с фиксированной пропускной способностью. Поэтому в начальной стадии атаки, когда администратор еще не успел определить признаки, отличающие пакеты атаки от пакетов легальных пользователей, файервол принципиально не может защитить сервер от атаки, пропуская весь трафик к серверу и тем самым блокируя его работу.

В сети провайдера облачных услуг отразить DDoS-атаку на ресурсы клиента принципиально проще. На рис. 30.8 показана достаточно типичная структура сети облачного провайдера с четырьмя центрами данных, рассредоточенными по географическим регионам, при этом в каждом центре имеется сервер с копией данных некоторого клиента. Для баланса нагрузки провайдер применяет маршрутизацию с произвольной (anycast) рассылкой (см. главу 14), в результате трафик запросов клиентов направляется ближайшему (относительно метрики маршрутизации) серверу.

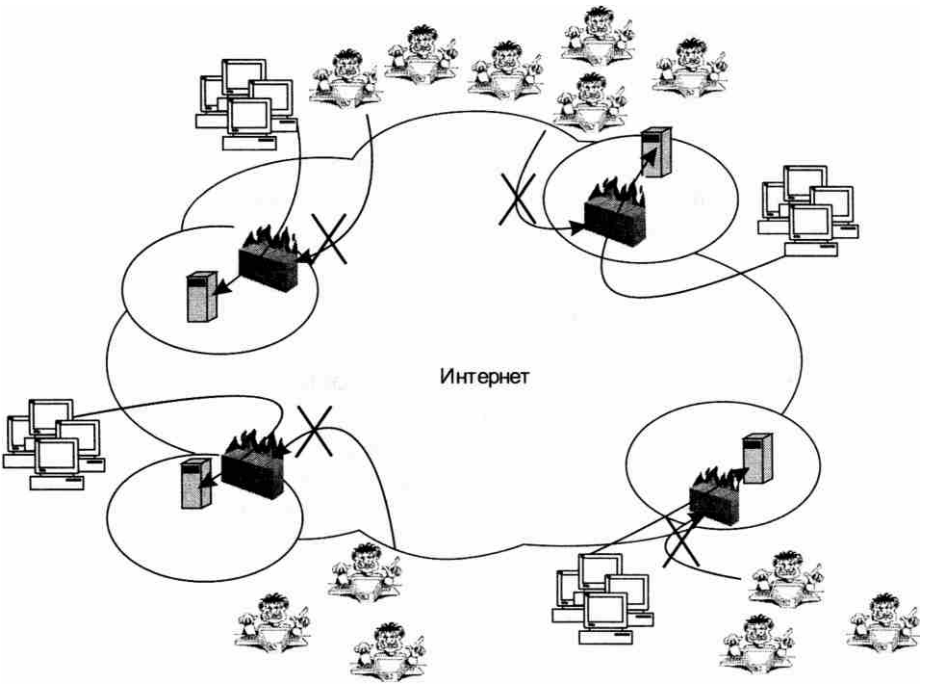


Рис. 30.8. Распределение и поглощение трафика DDoS-атаки инфраструктурой облачного провайдера

При возникновении DDoS-атаки трафик ботов злоумышленника также распределяется между серверами провайдера, как и трафик клиентов, поэтому атака не может быть такой же эффективной, как в случае единственного сервера и единственной линии доступа к нему. К тому же провайдер, как правило, поддерживает значительный запас пропускной способности линий доступа для успешного обслуживания пиков потребностей клиентов, поэтому «забить» линии доступа облачного провайдера злоумышленнику труднее. Можно сказать, что в начальный период DDoS-атаки инфраструктура облачного провайдера «впитывает» трафик атаки как губка без значительного ущерба для трафика легальных пользователей. А так как серверы провайдера защищены высокопроизводительными файерволами, то после обнаружения факта атаки и определения признаков, по которым можно отличить трафик атаки от трафика пользователей, администратор провайдера вносит соответствующие изменения в правила файерволов и они начинают блокировать трафик атаки. Внимательный читатель может заметить, что только что прочитанное пояснение вполне подходит к описанию DDoS-атак на корневые DNS-серверы, — и это неудивительно, так как механизм поглощения трафика атаки в обоих случаях один и тот же.

Квалификация персонала и качество платформы вычислений

Провайдер облачных услуг является крупной организацией (иначе он не сможет обеспечить масштабируемость, эластичность и некоторые другие свойства этой модели), и как

всякая крупная организация может себе позволить специализацию своих администраторов и операторов во всех областях обеспечения информационной безопасности. В результате специалисты провайдера получают большой опыт в распознавании всего спектра угроз, актуальных в настоящее время, а также в установке и конфигурировании средств отражения этих угроз, таких как файерволы, системы IDS/IPS, антивирусные системы и т. п., что, естественно, повышает безопасность данных клиентов облачных сервисов.

Важно также, что платформа вычислений (сетевая и компьютерная) может оказаться более качественной у облачного провайдера, чем у предприятия-клиента. Причиной тому является ее более *высокая степень однородности*, которая определяется тем, что провайдер оказывает одни и те же услуги большому количеству клиентов. Поэтому центры данных провайдера, как правило, строятся на одних и тех же моделях маршрутизаторов, коммутаторов, файерволов, серверов и гипервизоров виртуальных машин. Однородность упрощает управление и сокращает количество ошибок конфигурирования, а значит, делает более простым и эффективным обеспечение безопасности такой платформы.

Другая причина заключается в *масштабности* центров данных и сети провайдера облачных сервисов, что позволяет ему покупать для платформы самые совершенные и производительные компоненты защиты данных, например высокопроизводительный файервол, способный отфильтровывать пакеты интенсивных DoS-атак. Провайдеры облачных услуг стараются сертифицировать свои платформы по различным программам сертификации безопасности, чтобы завоевать доверие клиентов, — это также повышает вероятность того, что используемая платформа обладает высоким качеством.

Стоит ли обращаться к облачным сервисам?

После выяснения новых видов угроз, связанных с применением публичных облачных сервисов, у корпоративных специалистов по безопасности может возникнуть вопрос: а стоит ли овчинка выделки? Ответ неочевиден, но при его выборе нужно принимать во внимание не только тактические, но и стратегические соображения. А стратегические соображения говорят, что у облачных сред есть большое будущее, так что если сегодня они, возможно, еще не созрели для немедленного применения крупным предприятием с большим количеством чувствительных данных, то их, безусловно, нужно изучать и, возможно, опробовать, имея в виду потенциальную значимость для развития информационных технологий, и не только их. Соображения перспективности и значимости облачных сервисов должны учитываться администраторами безопасности и руководством предприятия при выработке политики безопасности в отношении использования этих сервисов.

Выводы

Уязвимости программного кода возникают в результате ошибок программирования, несовершенства операционной системы и защитных механизмов процессора.

Большой класс атак эксплуатирует возможность некорректного использования памяти в пределах адресного пространства отдельного процесса или в области памяти ядра операционной системы. Сюда относится переполнение буфера и переполнение стека.

Переполнение буфера является частным случаем уязвимостей, являющихся следствием слабого контроля вводимых данных. В более общем случае — любая не предусмотрен-

ная создателем программы форма вводимых данных может вызвать нарушение работы системы.

Многочисленная группа атак связана с внедрением в компьютеры вредоносных программ, к числу которых относятся троянские и шпионские программы, черви, вирусы, спам, логические бомбы и некоторые другие типы программ, нацеленные на нарушение безопасности. Троянские программы, или трояны, — это разновидность вредоносных программ, которые наносят ущерб системе, маскируясь под какие-либо полезные приложения. Сетевые черви — это программы, способные к самостоятельному распространению своих копий среди узлов в пределах локальной сети, а также по глобальным связям, перемещаясь от одного компьютера к другому без всякого участия в этом процессе пользователей сети. Вирус — это вредоносный программный фрагмент, который может внедряться в другие файлы.

Программная закладка — это встроенный в программное обеспечение объект, который при определенных условиях (входных данных) инициирует выполнение не описанных в документации функций, позволяющих осуществлять несанкционированные воздействия на информацию. Функции, описание которых отсутствует в документации, называют недекларированными возможностями.

Антивирусные программы для обнаружения вредоносных кодов в файлах, сообщениях электронной почты и HTML-страницах используют метод сигнатур. Сигнатура — это характерная последовательность программных кодов, которая с некой вероятностью идентифицирует вредоносный код.

Особенностью защиты веб-службы и электронной почты является то, что она требует решения двух достаточно независимых задач: обеспечения безопасности программных и аппаратных ресурсов компьютера, на котором эта служба выполняется, и обеспечения приватности того лица, которое этой службой пользуется.

Угрозу приватности несут службы фиксации событий веб-серверов и веб-браузеров, а также куки. Атаки на веб-службу и почту выражаются в нарушении конфиденциальности (прослушивании трафика), целостности (изменение передаваемой информации) и доступности (DoS-атаки на серверы).

Основным способом обеспечения безопасности данных, циркулирующих между веб-браузером и веб-сайтом, является использование безопасного протокола HTTPS.

Для снижения рисков почтовой службы применяются следующие подходы: совершенствование методов аутентификации отправителя (например, использование цифровых сертификатов), шифрование содержимого письма, защита метаданных пользователя.

Особыми видами атак, для которых почта служит инструментом, являются атаки с размещением вредоносного кода в почтовом приложении, а также спам.

Новая концепция организации вычислений, получившая название облачных, изменяет привычный мир пользователя, так как в соответствии с ней компьютер, который выполняет программу пользователя, находится где-то в «облаке» вычислительных ресурсов — процессоров, оперативной памяти, дисковых накопителей. Несмотря на то что облачные сервисы приято рассматривать как источник новых угроз безопасности, они могут существенно повысить информационную безопасность предприятия. Распределенная избыточная инфраструктура центров данных облачного провайдера позволяет эффективно бороться с DDoS-атаками.

Контрольные вопросы

1. Какую цель обычно преследует злоумышленник, используя эффект переполнения стека? Варианты ответов:
   а) испортить локальные переменные функции ядра ОС;
   б) поместить в стек вредоносный код и передать ему управление;
   в) исчерпать оперативную память компьютера.
2. Антивирусная программа, работающая по методу сигнатур, может:
   а) выявлять только известные вредоносные коды;
   б) выявлять только вирусы, но не черви;
   в) выявлять троянских коней;
   г) выявлять только новые виды вредоносных кодов.
3. Могут ли куки, переданные веб-сервром вашему веб-браузеру, заразить ваш компьютер вирусом?
4. С какой целью веб-сервер передает куки веб-браузеру клиента? Варианты ответов:
   а) для создания динамических веб-страниц;
   б) для сохранения состояния сеанса пользователя;
   в) для повышения защищенности сеанса.
5. Какое утверждение относительно протокола HTTPS является правильным? Варианты ответов:
   а) протокол HTTPS использует защищенный канал SSL для предотвращения атак на сеанс между веб-браузером и веб-сервером;
   б) протокол HTTPS не является протоколом в строгом смысле этого термина;
   в) протокол HTTPS использует цифровые сертификаты веб-браузера и веб-сервера для образования защищенного канала.
6. Что делает облачные вычисления более безопасными, чем традиционные? Варианты ответов:
   а) высокая квалификация специалистов предприятий, предоставляющих облачные сервисы;
   б) высокое качество вычислительной платформы;
   в) способность поглощать трафик DDoS;
   г) централизация вычислительных ресурсов.

Рекомендуемая и использованная литература

1. *Блэк Ю.* Сети ЭВМ: протоколы стандарты, интерфейсы / Перев. с англ. — М.: Мир, 1990.
2. *Стивен Браун.* Виртуальные частные сети. — М.: Лори, 2001.
3. *Шринивас Вегешна.* Качество обслуживания в сетях IP. — М.: Вильямс, 2003.
4. *Галатенко В.* Основы информационной безопасности. — М.: Бином. Лаборатория знаний, 2008.
5. *Гольдштейн Б. С., Пинчук А. В., Суховицкий А. Л.* IP-телефония, «Радио и связь», 2001.
6. *Денисьева О. М., Мирошников Д. Г.* Средства связи для «последней мили». — М.: Эко-Трендз, 1998.
7. *Аннабел З. Додд.* Мир телекоммуникаций: Обзор технологий и отрасли. — М.: ЗАО «Олимп-Бизнес», 2002.
8. *Дженнингс Ф.* Практическая передача данных: Модемы, сети и протоколы / Перев. с англ. — М.: Мир, 1989.
9. *Кейт Дж. Джонс.* Анти-хакер. Средства защиты компьютерных сетей: Справочник профессионала. 2003.
10. *Оливер Ибе.* Сети и удаленный доступ: Протоколы, проблемы, решения. М.: ДМК Пресс, 2002.
11. *Дуглас Э. Камер.* Сети TCP/IP. Т. 1. Принципы, протоколы и структура. — М.: Вильямс, 2003.
12. *Кеннеди Кларк, Кевин Гамильтон.* Принципы коммутации в локальных сетях Cisco. — М.: Вильямс, 2003.
13. *Куинн Л., Рассел Р.* Fast Ethernet. — BHV-Киев, 1998.
14. *Куроуз Дж., Росс К.* Компьютерные сети, 4-е изд. СПб.: Питер, 2004.
15. *Лапонина О. Р.* Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия: Курс лекций. — М.: Бином. Лаборатория знаний, 2009.
16. *Дилип Найк.* Стандарты и протоколы Интернета Channel Trading Ltd., 1999.
17. *Олифер В.* Направления развития средств безопасности предприятия. «Электроника», № 1, 2001.
18. *Олифер В. Г., Олифер Н. А.* Безопасность компьютерных сетей. — М.: Горячая линия–Телеком, 2014.
19. *Олифер В. Г., Олифер Н. А.* Новые технологии и оборудование IP-сетей. — СПб.: БХВ-Санкт-Петербург, 2000.
20. *Олифер В. Г., Олифер Н. А.* Сетевые операционные системы, 2-е изд. — СПб.: Питер, 2008.
21. *Олифер В. Г., Олифер Н. А.* Основы компьютерных сетей. — СПб.: Питер, 2009.
22. *Олифер В., Петрусов Д.* Внедрение услуг IP-телефонии в сети оператора связи // Аналитический и информационный журнал Документальная Электросвязь, № 8, январь 2002.

23. *Пятибратов А. П., Гудыно Л. П., Кириченко А. А.* Вычислительные системы, сети и телекоммуникации. — М.: Финансы и статистика, 2004.
24. *Фейт Сидни.* TCP/IP. Архитектура, протоколы, реализация. — М.: Лори, 2000.
25. *Сингх Саймон.* Книга шифров. Тайная история шифров и их расшифровки. — М.: АСТ, Астрель, 2007.
26. *Слепов Н. Н.* Синхронные цифровые сети SDH. — М.: Эко-Трендз, 1998.
27. *Марк Спортак, Френк Паппас и др.* Компьютерные сети и сетевые технологии. — М.: ТИД «ДС», 2002.
28. *Стерн, Монти.* Сети предприятий на основе Windows NT для профессионалов / Пер. с англ. — СПб.: Питер, 1999.
29. *Ричард Стивенс.* Протоколы TCP/IP: Практическое руководство. — СПб.: БХВ, 2003.
30. *Столингс В.* Передача данных, 4-е изд. — СПб.: Питер, 2004.
31. *Столингс В.* Современные компьютерные сети, 2-е изд. — СПб.: Питер, 2003.
32. *Эд Титтель, Стив Джеймс, Дэвид Пискителло, Лайза Пфайфер.* ISDN просто и доступно. — М.: Лори, 1999.
33. *Таненбаум Э.* Компьютерные сети. 4-е изд. — СПб.: Питер, 2002.
34. *Уолрэнд Дж.* Телекоммуникационные и компьютерные сети: Вводный курс. — М.: Постмаркет, 2001.
35. *Фред Халсалл.* Передача данных, сети компьютеров и взаимосвязь открытых систем. — М.: Радио и связь, 1995.
36. *Крейг Хант.* Персональные компьютеры в сетях TCP/IP / Перев. с англ. — BHV-Киев, 1997.
37. *Джон Чирилло.* Защита от хакеров — СПб.: Питер, 2002.
38. *Харрис Ш.* CISSP. Руководство для подготовки к экзамену, 2011.
39. *Щербо В. К.* Стандарты вычислительных сетей. Взаимосвязи сетей: Справочник. — М.: Кудиц-образ, 2000.
40. *Uyless Black.* Emerging Communications Technologies, 2/e, Prentice Hall Professional, 1997.
41. *Jerry H. Saltzer, Mike D. Schroeder.* «The protection of information in computer systems», 1975.
42. *Donn B. Parker.* Fighting Computer Crime: A New Framework for Protecting Information, 1998.
43. *Thomas R. Peltier.* Information Security Risk Analysis, Third Edition, 2010.
44. *Peyman Kabiri.* Privacy Intrusion Detection and Response, IGI Global, 2011.
45. *Michael G. Solomon.* Security Strategies in Windows Platforms and Applications — Jones & Bartlett Learning, 2010.
46. *Charles P. Pfleeger, Shari Lawrence.* Security in Computing, Fourth Edition — Prentice Hall, 2006.
47. *Wes Noonan; Ido Dubrawsky.* Firewall Fundamentals — Cisco Press, 2006.
48. *Josef Pieprzyk, Thomas Hardjono, Jennifer Seberry.* Fundamentals of Computer Security — Springer, 2010.
49. *Eric Cole.* Network Security Bible, 2nd Edition — John Wiley & Sons, 2009.
50. *Paul Kocher, Joshua Jaffe, Benjamin Jun* Introduction to Differential Power Analysis and Related Attacks, 1998.

Ответы

Глава 1

1. От вычислительной техники компьютерными сетями были унаследованы интеллектуальные возможности конечных узлов — компьютеров, а от телекоммуникационных сетей — методы передачи информации на большие расстояния.

2. Вычислительные ресурсы многотерминальных систем централизованы, а в компьютерной сети они распределены.

3. В эффективности передачи пульсирующего трафика.

Глава 2

1. Варианты а), в) и г).

2. Вариант в).

3. Вариант д).

4. Варианты а) и б).

5. Да.

Глава 3

1. Пульсирующий трафик.

2. Да.

3. Варианты а) и б).

4. Варианты а) и г).

5. Вариант в).

Глава 4

1. Да.

2. Все утверждения верны.

3. Да.

4. Три уровня.

Глава 5

1. Да.

2. Время сериализации и время распространения сигнала.

3. Да.

5. Нет.

Глава 6

1. Нет.
2. Вытеснение низкоприоритетного трафика.
4. Взвешенные очереди.
5. Маршрут.

Глава 7

1. Проводная и беспроводная.
2. Может, если они оцифрованы.
4. Варианты а) и в).

Глава 8

1. Вариант а).
3. Один символ кода, имеющий 12 состояний, передает 3,58 бита.
5. Варианты а) и б).

Глава 9

2. За счет отражения ионосферой Земли.
3. Туман, роса, дождь.
4. Вариант б).
5. Варианты а) и б).

Глава 10

1. За счет «плавающих» виртуальных контейнеров внутри кадра SDH.
2. Вариант б).
3. Потому что в этом режиме кадры полностью буферизуются.
4. И в сетях FDM, и в сетях DWDM используется частотное мультиплексирование.
5. Вариант а).

Глава 11

1. Вариант а).
2. Вариант б).
3. Вариант а).
4. Вариант а).
5. Вариант в).

Глава 12

1. Варианты а) и в).
2. Вариант б).
3. Варианты а) и б).
4. Да.
5. Нет.

Глава 13

1. Нет, корневой мост не имеет корневых портов.
2. Да.
3. Варианты б) и в).
4. Варианты а), б) и в).
5. Да.

Глава 14

2. Номер подсети 108.5.16.0. Для нумерации интерфейсов в данной сети может быть использовано 12 бит, то есть 4096 значений. Но так как двоичные значения, состоящие из одних нулей и одних единиц, зарезервированы, то в сети не может быть более 4094 узлов.
3. Об IP-адресах узлов ничего определенного сказать нельзя.
4. Вариант в).
5. Максимум можно организовать 16 385 подсетей. При этом маска должна иметь значение 255.255.255.252 (детали см. на сайте www.olifer.co.uk).

Глава 15

1. Да.
2. Нет.
3. Вариант г).
4. Вариант в).
5. Вариант в).

Глава 16

2. Вариант в).
3. Вариант а).
4. Объем полученных данных составляет 165 005 байт.
5. Вариант б).

Глава 17

1. Варианты а) и г)

2. Да, в сети можно обойтись без протоколов маршрутизации, если создавать таблицы маршрутизации вручную.

3. Вариант в).

4. Варианты а), б) и в).

Глава 18

1. Вариант в).

2. Варианты а) и в).

3. Варианты а) и в).

4. Нет.

5. Варианты а) и б).

Глава 19

1. Вариант в).

2. Да, были помечены кадры 6 и 7, так как согласованная величина пульсации равна:

$$CIR \times T = 51200 \text{ бит} = 6400 \text{ байт},$$

Используется почтовым клиентом для передачи письма на сервер	SMTP
Используется почтовым клиентом для получения письма с сервера	POP3, IMAP
При получении почты письмо перемещается с сервера на клиент	POP3
При получении почты письмо копируется с сервера на клиент	IMAP

3.

Путь к объекту	/mobile/web/versions.shtml
DNS-имя сервера	www.bbc.co.uk
URL-имя	http://www.bbc.co.uk/mobile/web/versions.shtml
Тип протокола доступа	http://

4. Варианты б), в) и е).

5. Варианты б), г) и д).

Глава 24

1. Модель с запоминанием состояния.

2. Варианты а), б) и г).

3. Каждый пользователь видит на экране не только свои исправления, но и исправления, сделанные другим пользователем.

Глава 25

1. Варианты а) и г).
2. Вариант б).
3. Вариант б).
4. Варианты а) и д).

Глава 26

1. Необходимым условием осуществления *атаки* является наличие *уязвимости* и направленной на ее использование *угрозы*.
2. Варианты б), в), г) и д).
3. Варианты б) и в).
4. Варианты б), в) и г).
5. Варианты а), б) и г).

Глава 27

1. Варианты б), в) и г).
2. Варианты а), б) и в).
3. Вариант в).
4. Варианты а) и в).
5. Вариант б).
6. Варианты б) и в).

Глава 28

1. Варианты в) и г).
2. Варианты а), в) и г).
3. Варианты в) и г).
4. Варианты а) и б).
5. Да.

Глава 29

1. Варианты а), в) и г).
2. Варианты а), б), в) и г).
3. Вариант б).
4. Вариант в).
5. Варианты а) и в).

Глава 30

1. Вариант б).
2. Вариант а).
3. Нет.
4. Вариант б).
5. Варианты а), б) и в).
6. Варианты а), б) и в).

Алфавитный указатель

Б

В

Е

З

И

К

Л

М

П

С

Т

В. Олифер, Н. Олифер

Компьютерные сети. Принципы, технологии, протоколы
Учебник для вузов

5-е издание

Заведующая редакцией	*Ю. Сергиенко*
Ведущий редактор	*Н. Римицан*
Литературный редактор	*А. Жданов*
Художник	*С. Заматевская*
Корректоры	*С. Беляева, Н. Викторова, М. Одинокова*
Верстка	*Л. Соловьева*

ООО «Питер Пресс», 192102, Санкт-Петербург, ул. Андреевская (д. Волкова), 3, литер А, пом. 7Н.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12.000 — Книги печатные профессиональные, технические и научные.

Подписано в печать 30.09.15. Формат 70×100/16. Бумага писчая. Усл. п. л. 79,980. Тираж 3500. Заказ № ВЗК-04894-15.

Отпечатано в АО «Первая Образцовая типография», филиал «Дом печати — ВЯТКА»
610033, г. Киров, ул. Московская, 122

Учебник для вузов

Компьютерные сети

Профессиональные биографии Виктора и Натальи Олифер очень похожи. Оба они получили свое первое высшее образование в МВТУ им. Н. Э. Баумана (специальность «Электронные вычислительные машины»), а второе — в МГУ им. М. В. Ломоносова (специальность «Прикладная математика»). После защиты диссертации каждый из них совмещал преподавание в вузах с научно-исследовательской работой. В 1995 году Наталья и Виктор стали читать лекции по сетевым технологиям в Центре информационных технологий при МГУ. Ими были разработаны несколько авторских курсов, которые и составили в дальнейшем основу для написания учебников «Компьютерные сети», «Сетевые операционные системы», «Безопасность компьютерных сетей», а также книги "Computer Networks: Principles, Technologies and Protocols for Network Design".
В настоящее время Наталья Олифер работает независимым консультантом в области сетевых технологий, а Виктор Олифер принимает участие в научно-технических разработках, таких как проект Janet – создание объединяющей сети кампусов университетов и исследовательских центров Великобритании, и панъевропейские проекты GEANT2, GEANT3 и GEANT4.

Пятое издание одного из лучших российских учебников по сетевым технологиям, переведенного на английский, испанский, португальский и китайский языки, отражает те изменения, которые произошли в области компьютерных сетей за 6 лет, прошедших со времени подготовки предыдущего издания: преодоление локальными и глобальными сетями рубежа скорости в 100 Гбит/с и освоение терабитных скоростей; повышение эффективности и гибкости первичных оптических сетей за счет появления реконфигурируемых мультиплексоров ввода-вывода (ROADM) и применения суперканалов DWDM, работающих на основе гибкого частотного плана; развитие техники виртуализации сетевых функций и услуг, приведшей к распространению облачных сервисов; выход на первый план проблем безопасности.

Издание предназначено для студентов, аспирантов и технических специалистов, которые хотели бы получить базовые знания о принципах построения компьютерных сетей, понять особенности традиционных и перспективных технологий локальных и глобальных сетей, изучить способы создания крупных составных сетей и управления такими сетями.

Рекомендовано Министерством образования и науки Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению «Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем».

ПИТЕР®

Заказ книг:
Санкт-Петербург
тел.: (812) 703-73-74, postbook@piter.com

www.piter.com — каталог книг и интернет-магазин

ISBN 978-5-496-01967-5