Высшее профессиональное образование

Учебник

В. В. Платонов

ПРОГРАММНО-АППАРАТНЫЕ СРЕДСТВА ЗАЩИТЫ ИНФОРМАЦИИ

БАКАЛАВРИАТ

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

ACADEMIA

ПРОГРАММНО-АППАРАТНЫЕ СРЕДСТВА ЗАЩИТЫ ИНФОРМАЦИИ

УДК 004.56(075.8)
ББК 32.81я73
П375

Рецензент —
зав. кафедрой «Информатика и информационная безопасность»
Санкт-Петербургского государственного университета путей сообщения,
д-р техн. наук, проф. *А. А. Корниенко*

Платонов В. В.

П375 Программно-аппаратные средства защиты информации : учебник для студ. учреждений высш. проф. образования / В. В. Платонов. — М. : Издательский центр «Академия», 2013. — 336 с. — (Сер. Бакалавриат).

ISBN 978-5-7695-9327-7

Учебник создан в соответствии с Федеральным государственным образовательным стандартом по направлению подготовки 090900 «Информационная безопасность» («квалификация «бакалавр»).

Показано обеспечение безопасности межсетевого взаимодействия. Рассмотрены основные виды вредоносных программ, удаленные сетевые атаки и организация защиты от них. Изложены методы описания атак и основные тенденции их развития. Описаны основные технологии межсетевых экранов, их оценка и тестирование. Проанализированы методы построения систем обнаружения вторжений. Рассмотрены проблемы защиты при организации удаленного доступа, построение и функционирование виртуальных ведомственных сетей (VPN), а также основные отечественные средства для их построения.

Для студентов учреждений высшего профессионального образования.

УДК 004.56(075.8)
ББК 32.81я73

ISBN 978-5-7695-9327-7

ПРЕДИСЛОВИЕ

> Я пришел к выводу, что наилучшими экспертами в области безопасности являются люди, исследующие несовершенства защитных мер.
>
> *Б. Шнайер. Секреты и ложь*

Современный мир невозможно представить без средств коммуникаций и вычислительной техники, в которых главенствующую роль играет программное обеспечение. Информационные технологии прогрессируют очень быстро, охватывая все более широкие области человеческой деятельности. Поэтому безопасность информационных технологий является одним из важнейших аспектов обеспечения их функционирования.

Коммуникации и информационные технологии слились воедино. Во многом этому способствовало использование сети Интернет, которая стала глобальной и всеобщей средой коммуникации. Стеку протоколов TCP/IP, разработанному более 30 лет назад, несмотря на его недостатки, удалось покорить весь мир. Рост числа компьютеров и компьютерных сетей, все более широкое использование сетевых технологий и технологии Интернета не только значительно расширили географию пользователей, их возможности по общению друг с другом, но и увеличили возможность реализации сетевых бизнес-процессов. Организации, компании и рядовые пользователи получили возможность использовать технологии Интернета в своей повседневной деятельности. Этому способствует развитие существующих служб и появление новых, востребованных мировым сообществом. Кроме увеличивающихся возможностей использование интернет-технологий значительно увеличивает и риск потери данных, потери репутации и просто финансовые потери. Во многом это обусловлено тем, что изначально технологии Интернета не были предназначены для обеспечения безопасности функционирования, кроме того, первые пользователи Интернета главной задачей считали обеспечение возможности надежной связи друг с другом.

Статистика показывает, что с каждым годом растет финансовый ущерб, наносимый компьютерными преступниками. Рост числа пользователей, недостатки в программном обеспечении пользователей, наличие свободного доступа к зловредным программам, а также практическая ненаказуемость совершения проступков при-

вели к тому, что организациям, компаниям и рядовым пользователям приходится уделять внимание и время обеспечению защиты.

Современные распределенные компьютерные системы не могут быть защищены только использованием организационных мер и средств физической защиты. Для безопасности функционирования информационных технологий необходимо использовать механизмы и средства сетевой защиты, которые обеспечивают конфиденциальность, целостность и доступность компьютерных систем, программного обеспечения и данных.

В последнее десятилетие сформировался рынок средств обеспечения информационной безопасности, все большее число различных организаций подключается к решению насущных задач обеспечения защиты. Решение этих задач осложняется рядом причин. Среди них необходимо отметить следующие: отсутствие единой терминологии и единой теории обеспечения защиты, использование, как правило, программных средств зарубежных производителей, высокую стоимость качественных средств защиты. Реализация защиты информации в современных корпоративных сетях опирается на использование средств защиты, реализованных программными, аппаратными и программно-аппаратными методами. Количество изданий и статей, посвященных различным вопросам обеспечения информационной безопасности, увеличивается с каждым годом.

В настоящем учебнике рассмотрены основные технологии, используемые при построении систем защиты корпоративных сетей. «Безопасность — это процесс, а не продукт»[1], — отметил Б.Шнайер в своей книге «Секреты и ложь». Область задач обеспечения информационной безопасности достаточно широка. Вне рамок рассмотрения остались многие важные вопросы, например, защита операционных систем, защита баз данных, защита информации от утечки по техническим каналам, защита беспроводных сетей, а также вопросы криптографической защиты. Рассмотрены подходы и технологии, лежащие в основе построения систем защиты.

В настоящем издании использованы материалы лекций дисциплин, которые читаются на кафедре «Информационная безопасность компьютерных систем» факультета технической кибернетики Санкт-Петербургского государственного политехнического университета.

Учебник состоит из шести глав. В гл. 1 рассмотрены проблемы обеспечения безопасности межсетевого взаимодействия, приводится системная классификация угроз. Изложены основные вопросы обеспечения информационной безопасности и обобщенная схема управления безопасностью, вопросы построения политики безопасности организации и существующие шаблоны политик. Для сетевой политики безопасности рассмотрены сетевые периметры организа-

[1] *Шнайер Б.* Секреты и ложь. Безопасность данных в цифровом мире / Б. Шнайер. — СПб. : Питер, 2003. — 368 с.

ции и элементы построения эшелонированной защиты. В качестве одной из важнейших задач управления информационной безопасностью рассмотрен подход к управлению рисками, включая вопросы оценки риска и методику его уменьшения. Изложены основные понятия аудита и его значение в обеспечении информационной безопасности.

В гл. 2 рассмотрены основные виды вредоносного программного обеспечения, их особенности и методы защиты.

В гл. 3 рассмотрена основная угроза межсетевого взаимодействия — удаленные сетевые атаки. Приведены элементы терминологии и рассмотрены примеры некоторых атак. Основное внимание уделено существующим подходам к классификации атак и рассмотрено построение онтологии удаленных сетевых атак. Приведен подход и примеры оценивания степени серьезности атак.

В гл. 4 рассмотрены основные этапы становления технологий межсетевых экранов. Технологии проиллюстрированы примерами списков контроля доступа для маршрутизаторов компании Cisco. Приведены методы обхода межсетевых экранов, а также основные подходы к тестированию экранов.

В гл. 5 рассмотрены системы обнаружения вторжений, приводится их классификация. Основное внимание уделено анализу перспективных подходов к обнаружению, использующих технологии искусственного интеллекта. Рассмотрены методы обхода (обмана) систем обнаружения вторжений и существующие методики тестирования систем обнаружения, подходы к построению систем предотвращения вторжений.

В гл. 6 рассмотрены общие вопросы туннелирования и принципы построения виртуальных частных сетей. Проанализированы основные протоколы организации виртуальных частных сетей на канальном, сетевом и транспортном уровнях.

В Приложении даны основные используемые понятия (определения), приводимые в стандартах Российской Федерации, и списки использованных RFC (Request for Comments).

Глава 1
ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ МЕЖСЕТЕВОГО ВЗАИМОДЕЙСТВИЯ

> И вправду, план был замечательный — такой простой и ясный. Одно только плохо: Алиса не имела ни малейшего представления о том, как все это осуществить.
>
> *Л.Кэррол. Алиса в стране чудес*

Современные вычислительные сети организаций представляют собой сложные системы, состоящие из множества компонентов. Среди этого множества компонентов можно выделить разнообразные компьютеры, системное и прикладное программное обеспечение (ПО) этих компьютеров, сетевые адаптеры, концентраторы, коммутаторы, маршрутизаторы и соединительные (кабельные) системы.

Широкое использование Интернета и интернет-технологий привело к качественному изменению вычислительных сетей. Если ранее Интернет использовался в основном в качестве среды передачи, то в настоящее время Интернет становится не только средством интерактивного взаимодействия людей, но и средством ведения деловых операций организаций, реальным средством проведения бизнес-операций.

Популярность IP-технологий объясняется их объективными достоинствами. К числу таких достоинств можно отнести относительную простоту основополагающих принципов технологии. Одним из таких принципов является открытость, что выражается свободным обсуждением, исследованием и тестированием новых протоколов стека TCP/IP в рамках не только рабочих групп комитета Internet Engineering Task Force (IETF), но и всего мирового сообщества. Разрабатываемые и предлагаемые стандарты и спецификации доступны практически всем пользователям Интернет. Открытость технологии позволяет обеспечить относительно простую интеграцию в IP-сети других технологий, что значительно увеличивает области применения Интернета.

Другим достоинством IP-технологий является масштабируемость, которая была заложена уже при разработке Интернета. Иерархически организованный стек TCP/IP позволяет наращивать сети организаций в достаточно больших пределах.

Эти и другие достоинства обеспечили на настоящий момент широкое применение IP-технологий. Технологии, которые привели к

успеху Интернета, оказались чрезвычайно перспективными и для внутренних сетей организаций — сетей интранет (intranet).

Корпоративная сеть (интранет) — это сеть на уровне компании, в которой используются программные средства, основанные на стеке протоколов TCP/IP.

Под *экстранет-сетями* понимается интранет-сеть, подключенная к Интернету, т.е. это сеть типа интранет, но санкционирующая доступ к ее ресурсам определенной категории пользователей, наделенной соответствующими полномочиями.

Поскольку в дальнейшем будут рассматриваться средства защиты, то все сети представляются как локальные сети, подключенные к Интернету. При этом рассмотрении не важно, используется ли в данной сети Web-технология, поэтому далее будем называть такие сети корпоративными.

Главные особенности корпоративных сетей — глобальность связей, масштабность и гетерогенность — представляют и повышенную опасность для выполнения ими своих функциональных задач. Поскольку протоколы семейства TCP/IP разработаны достаточно давно, когда проблема безопасности еще не стояла так остро, как сейчас, то они, в первую очередь, разрабатывались как функциональные и легко переносимые, что помогло распространиться стеку TCP/IP на множество компьютерных платформ. Кроме того, в настоящее время при использовании Интернета в распоряжении злоумышленников появляются многочисленные средства и методы проникновения в корпоративные сети.

1.1. Основы сетевого и межсетевого взаимодействия

Под *межсетевым взаимодействием* понимается взаимодействие двух локальных сетей, при котором они функционируют как самостоятельные единицы объединенной сети. Под взаимодействием сетей понимают методы расширения, сегментации и объединения локальных сетей таким образом, чтобы общая пропускная способность была как можно выше. В качестве устройств межсетевого взаимодействия выступают повторители, мосты, коммутаторы, маршрутизаторы и шлюзы. Кроме того, под межсетевым взаимодействием понимается совокупность протоколов, которая позволяет организовать обмен данными между различными сетями. Одним из наиболее употребительных наборов протоколов стало семейство протоколов TCP/IP, разработанное по заказу Министерства обороны США.

Протоколы TCP/IP были разработаны по инициативе Министерства обороны США для сети Arpanet — глобальной сети передачи пакетов, которая впервые была продемонстрирована в 1972 г. В на-

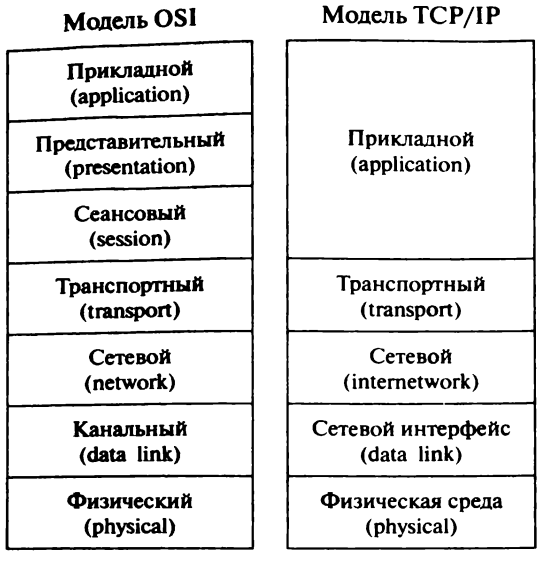


Рис. 1.1. Уровневые структуры стеков протоколов

стоящее время Arpanet является частью глобальной сети, известной как Интернет.

Громадное распространение Интернета привело к тому, что сетевой стек протоколов TCP/IP стал практически основным при организации межсетевого взаимодействия. Разработанная в конце 70-х годов XX в. совокупность протоколов опиралась на уровневую структуру, которая послужила основой для последующих разработок модели Open System Interconnection (рис. 1.1).

Технология данного стека протоколов оказалась настолько удобной, что ее стали использовать не только для работы в Интернете, но и при организации работы в корпоративных сетях.

В связи с гигантским ростом численности хостов, подключенных к Интернету, и ростом числа компаний, использующих технологии Интернета для ведения своего бизнеса, значительно увеличилось число инцидентов, связанных с информационной безопасностью (ИБ). Данные CERT (Computer Emergency Response Team) показывают, что число зарегистрированных инцидентов постоянно увеличивается (рис. 1.2).

Начиная с 2004 г. этот рост стал еще более значительным, в связи с чем CERT перестала публиковать итоговые данные. Количество инцидентов тесно связано с количеством обнаруженных уязвимостей. Сводные данные базы данных уязвимостей США (National Vulnerability Database, http://nvd.nist.gov) за последние 12 лет приведены на рис. 1.3.

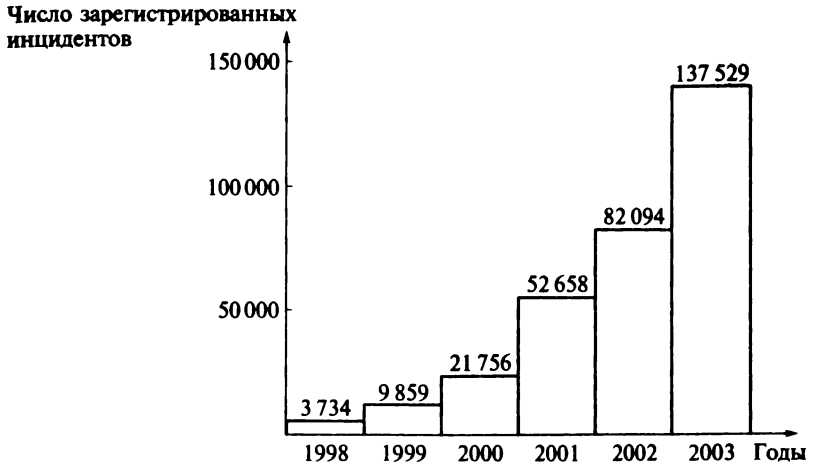


Рис. 1.2. Число зарегистрированных CERT инцидентов

Под *уязвимостью* (vulnerability) информационной системы понимается любая характеристика, использование которой нарушителем может привести к реализации угрозы.

Угрозой (threat) информационной системе называется потенциально возможное событие, действие, процесс или явление, которое может вызвать нанесение ущерба (материального, морального или иного) ресурсам системы.

К настоящему времени известно большое количество разноплановых угроз различного происхождения, таящих в себе различную опасность для информации. Системная классификация угроз приведена в табл. 1.1.

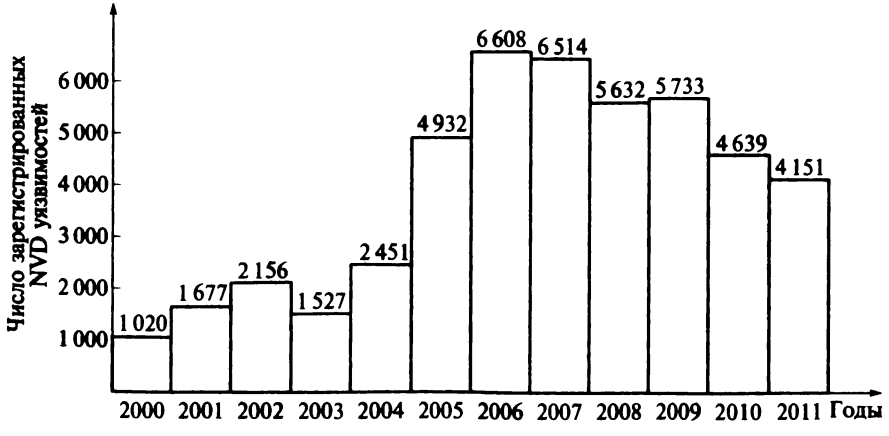


Рис. 1.3. Число зарегистрированных NVD уязвимостей

Таблица 1.1. **Системная классификация угроз**

Параметры классификации	Значения параметров	Содержание значения
Виды угроз	Физическая целостность	Уничтожение (искажение)
	Логическая структура	Искажение структуры
	Содержание	Несанкционированная модификация
	Конфиденциальность	Несанкционированное получение, утечка информации
	Право собственности	Присвоение чужого труда
Происхождение угроз	Случайное	Отказы, сбои, ошибки Стихийные бедствия Побочные влияния
	Преднамеренное	Злоумышленные действия людей
Предпосылки появления угроз	Объективное	Количественная и качественная недостаточность элементов системы
	Субъективное	Промышленный шпионаж, недобросовестные сотрудники, криминальные и хулиганствующие элементы, службы других государств
Источники угроз	Люди	Пользователи, персонал, посторонние люди
	Технические устройства	Регистрации, ввода, обработки, хранения, передачи и выдачи
	Модели, алгоритмы, программы	Общего назначения, прикладные, вспомогательные
	Технологические схемы обработки данных	Ручные, интерактивные, внутримашинные, сетевые
	Внешняя среда	Состояние среды, побочные шумы, побочные сигналы

Виды угроз — это основополагающий параметр, определяющий целевую направленность защиты информации.

Под случайным понимается такое происхождение угроз, которое обусловливается спонтанными и не зависящими от воли людей об-

стоятельствами, возникающими в системе обработки данных в процессе ее функционирования. Наиболее известными событиями данного плана являются отказы, сбои, ошибки, стихийные бедствия и побочные влияния:

• отказ — нарушение работоспособности какого-либо элемента системы, приводящее к невозможности выполнения им основных своих функций;

• сбой — временное нарушение работоспособности какого-либо элемента системы, следствием чего может быть неправильное выполнение им в этот момент своей функции;

• ошибка — неправильное (разовое или систематическое) выполнение элементом одной или нескольких функций, происходящее вследствие специфического (постоянного или временного) его состояния;

• побочное влияние — негативное воздействие на систему в целом или отдельные ее элементы, оказываемое какими-либо явлениями, происходящими внутри системы или во внешней среде.

Преднамеренное происхождение угрозы обусловливается злоумышленными действиями людей, осуществляемыми в целях реализации одного или нескольких видов угроз.

Отмечены две разновидности предпосылок появления угроз: объективные (количественная или качественная недостаточность элементов системы) и субъективные (деятельность разведывательных служб иностранных государств, промышленный шпионаж, деятельность криминальных и хулиганствующих элементов, злоумышленные действия недобросовестных сотрудников системы). Перечисленные разновидности предпосылок интерпретируются следующим образом:

• количественная недостаточность — физическая нехватка одного или нескольких элементов системы обработки данных, вызывающая нарушения технологического процесса обработки и (или) перегрузку имеющихся элементов;

• качественная недостаточность — несовершенство конструкции (организации) элементов системы, в силу чего могут появляться возможности для случайного или преднамеренного негативного воздействия на обрабатываемую или хранимую информацию;

• деятельность разведывательных служб иностранных государств — специально организуемая деятельность государственных органов, профессионально ориентированных на добывание необходимой информации всеми доступными способами и средствами;

• промышленный шпионаж — негласная деятельность организации (ее представителей) по добыванию информации, специально охраняемой от несанкционированной ее утечки или похищения, а также по созданию для себя благоприятных условий в целях получения максимальной выгоды;

• действия криминальных и хулиганствующих элементов — хищение информации или компьютерных программ в целях наживы или их разрушение в интересах конкурентов;

• злоумышленные действия недобросовестных сотрудников — хи-ение (копирование) или уничтожение информационных массивов (или) программ по эгоистическим или корыстным мотивам.

Источниками угроз являются люди, технические устройства, про-аммы и алгоритмы, технологические схемы обработки данных и ешняя среда:

• люди — персонал, пользователи и посторонние лица, которые огут взаимодействовать с ресурсами и данными организации непо-едственно с рабочих мест и удаленно, используя сетевое взаимо-йствие;

• технические средства — непосредственно связанные с обработ-ой, хранением и передачей информации (например, средства реги-рации данных, средства ввода и т.д.), и вспомогательные (напри-р, средства электропитания, кондиционирования и т.д.);

• модели, алгоритмы и программы — эту группу источников рас-атривают как недостатки проектирования, реализации и конфигу-ации (эксплуатации) и называют недостатками программного обе-ечения (общего назначения, прикладного и вспомогательного);

• технологическая схема обработки данных — выделяют ручные, нтерактивные, внутримашинные и сетевые технологические схемы бработки;

• внешняя среда — выделяют состояние среды (возможность по-аров, землетрясений и т.п.), побочные шумы (особенно опасные ри передаче данных) и побочные сигналы (например, электромаг-итное излучение аппаратуры).

Основными причинами утечки информации являются:

• несоблюдение персоналом норм, требований, правил эксплуа-ации;

• ошибки в проектировании системы и систем защиты;

• ведение противостоящей стороной технической и агентурной азведок.

Несоблюдение персоналом норм, требований, правил эксплуа-ации может быть как умышленным, так и непреднамеренным. От едения противостоящей стороной агентурной разведки этот слу-ай отличает то, что в данном случае лицом, совершающим несанк-ионированные действия, двигают личные побудительные мотивы. ричины утечки информации достаточно тесно связаны с видами ечки информации. Рассматриваются три вида утечки информа-ии:

• разглашение;

• несанкционированный доступ к информации;

• получение защищаемой информации разведками (как отече-венными, так и иностранными).

Под разглашением информации понимается несанкци-ированное доведение защищаемой информации до потребителей, е имеющих права доступа к защищаемой информации.

Под несанкционированным доступом понимается получение защищаемой информации заинтересованным субъектом с нарушением установленных правовыми документами или собственником, владельцем информации прав или правил доступа к защищаемой информации. При этом заинтересованным субъектом, осуществляющим несанкционированный доступ к информации, может быть государство, юридическое лицо, группа физических лиц (в том числе общественная организация), отдельное физическое лицо.

Получение защищаемой информации разведками может осуществляться с помощью технических средств (техническая разведка) или агентурными методами (агентурная разведка).

Канал утечки информации — совокупность источника информации, материального носителя или среды распространения несущего указанную информацию сигнала и средства выделения информации из сигнала или носителя.

Одним из основных свойств канала является месторасположение средства выделения информации из сигнала или носителя, которое может располагаться в пределах контролируемой зоны, охватывающей систему, или вне ее.

Далее будем рассматривать только угрозы, связанные с межсетевым взаимодействием.

1.2. Информационная безопасность

Начиная с середины 90-х годов XX в. стало ясно, что Интернет может использоваться не только как альтернативный канал получения информации, но и как экономически оправданная альтернатива. Развитие компьютерной техники, средств телекоммуникаций, миниатюризация аппаратуры привели к развитию информационной инфраструктуры нового поколения. Конкуренция и высокая инвестиционная привлекательность обеспечили лавинообразный рост числа предприятий и организаций, использующих инфраструктуру и технологии Интернета как каналы получения информации, необходимой для экономической деятельности.

Под *информационной безопасностью* понимается защищенность информации и поддерживающей инфраструктуры от случайных и преднамеренных воздействий естественного или искусственного характера, чреватых нанесением ущерба владельцам или пользователям информации и поддерживающей инфраструктуры.

Вопросы обеспечения информационной безопасности исследуются в разных странах достаточно давно. В настоящее время сложилась общепринятая точка зрения на концептуальные основы ИБ. Суть ее заключается в том, что подход к обеспечению ИБ должен быть ком-

плексным, сочетающим различные меры обеспечения безопасности на следующих уровнях:

- законодательный (законы, нормативные акты, стандарты);
- административный (действия общего характера, предпринимаемые руководством организации);
- процедурный (меры безопасности, реализуемые персоналом);
- программно-технический (конкретные технические меры).

При обеспечении ИБ существует два аспекта: *формальный* — определение критериев, которым должны соответствовать защищенные информационные технологии, и *практический* — определение конкретного комплекса мер безопасности применительно к рассматриваемой информационной технологии.

Независимо от размеров организации и специфики ее информационной системы работы по обеспечению режима ИБ в том или ином виде должны включать в себя следующие этапы:

- определение политики ИБ;
- определение сферы (границ) системы управления информационной безопасностью и конкретизация целей ее создания;
- оценка рисков;
- управление рисками;
- выбор контрмер, обеспечивающих режим ИБ;
- аудит системы управления ИБ.

Сущность информационной безопасности — защита объектов информации организации и бизнес-процессов, которые они поддерживают, в контексте:

- конфиденциальности — информация доступна только тем, кто авторизован для доступа;
- целостности — точность и полнота информации и методов обработки гарантирована;
- доступности — информация и ассоциированные объекты доступны по требованию авторизованных пользователей.

Международным сообществом установлено девять принципов обеспечения безопасности информационных систем (ИС) и сетей:

1) информированность (awareness) — участники должны сознавать необходимость безопасности информационных систем и сетей и то, что можно сделать для усиления безопасности;

2) ответственность (responsibility) — все участники ответственны за безопасность ИС и сетей;

3) реагирование (response) — участники должны действовать одновременно и совместно для защиты, обнаружения и реакции на инциденты безопасности;

4) этика (ethics) — участники должны соблюдать законные интересы друг друга;

5) демократия (democracy) — безопасность ИС и сетей должна быть совместима с главными ценностями демократического общества;

Рис. 1.4. Схема управления информационной безопасностью

6) оценка риска (risk assessment) — участники должны проводить оценку риска;

7) планирование (проектирование) и применение безопасности (security design and implementation) — участники должны включать механизмы обеспечения безопасности в информационные системы и сети;

8) управление безопасностью (security management) — участники должны использовать исчерпывающий подход к управлению безопасностью;

9) переоценка (reassessment) — участники должны пересматривать и переоценивать безопасность ИС и сетей и проводить соответствующие модификации политики безопасности, практики, оценок и процедур.

Защита информационных объектов является важнейшей задачей организации.

Управление рисками ИБ выступает интегральной частью процесса управления рисками организации. Схема управления ИБ приведена на рис. 1.4.

Основополагающим элементом защиты является определение политики безопасности защищаемой системы.

1.3. Политика безопасности

Политика информационной безопасности (или политика безопасности) является планом высокого уровня, в котором описываются цели и задачи организации, а также мероприятия в сфере обеспечения безопасности. Политика описывает безопасность в обобщенных терминах без специфических деталей. Она обеспечивает планирование всей программы обеспечения безопасности. Политика информационной безопасности должна обеспечить защиту выполнения задач организации или защиту делового процесса.

Средствами обеспечения делового процесса (бизнес-процесса) являются аппаратные средства и программное обеспечение, которые должны быть охвачены политикой безопасности (ПБ). Поэтому в качестве основной задачи необходимо предусмотреть полную инвентаризацию системы, включая карту сети. При составлении карты сети необходимо определить потоки информации в каждой системе. Схема информационных потоков может показать, насколько потоки информации обеспечивают бизнес-процессы, а также показать области, в которых важно обеспечить защиту информации, и принять дополнительные меры для обеспечения живучести. Кроме того, с помощью этой схемы можно определить места, в которых должна обрабатываться информация, как эта информация должна храниться, регистрироваться, дублироваться, перемещаться и контролироваться.

Инвентаризация, кроме аппаратных и программных средств, должна охватывать и некомпьютерные ресурсы, такие как программная документация, документация на аппаратуру, технологическая документация и т.д. Эти документы могут содержать информацию относительно особенностей организации бизнеса, а также могут показать области, которые могут быть использованы нарушителями.

Определение политики ИБ должно сводиться к следующим практическим шагам.

1. Определение используемых руководящих документов и стандартов в области ИБ, а также основных положений политики ИБ, в том числе:
   - управление доступом к средствам вычислительной техники (СВТ), программам и данным;
   - антивирусную защиту;
   - вопросы резервного копирования;
   - проведение ремонтных и восстановительных работ;
   - информирование об инцидентах в области ИБ.
2. Определение подходов к управлению рисками: является ли достаточным базовый уровень защищенности или требуется проводить полный вариант анализа рисков.
3. Определение требований к режиму информационной безопасности.
4. Структуризация контрмер по уровням.

5. Определения порядка сертификации на соответствие стандартам в области ИБ.

6. Определение периодичности проведения совещаний по тематике ИБ на уровне руководства, включая периодический пересмотр положений политики ИБ, а также порядок обучения всех категорий пользователей информационной системы по вопросам ИБ.

Реальная политика безопасности организации может включать в себя следующие разделы:

- общие положения;
- политика управления паролями;
- идентификация пользователей;
- полномочия пользователей;
- защита информационных ресурсов организации от компьютерных вирусов;
- правила установки и контроля сетевых соединений;
- правила политики безопасности по работе с системой электронной почты;
- правила обеспечения безопасности информационных ресурсов;
- обязанности пользователей по выполнению правил ПБ и т.д.

Политика безопасности не должна быть «мертвым» документом. Правила должны изменяться и развиваться по мере развития самой организации, появления новых технологий, систем и проектов. Для этого необходимо периодически пересматривать правила. Одним из методов пересмотра политики безопасности, в частности, является аудит информационных систем. Поэтому можно говорить, что политика безопасности организации и, естественно, политика инфор-

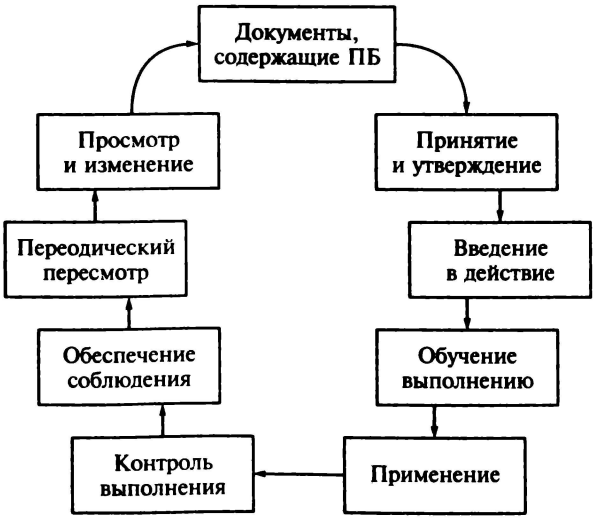


Рис. 1.5. Жизненный цикл политики безопасности

мационной безопасности имеют свой жизненный цикл. Жизненный цикл политики безопасности приведен на рис. 1.5.

Определенных сроков того, как часто надо пересматривать правила, не существует. Однако рекомендуется, чтобы этот срок составлял от шести месяцев до одного года.

После разработки и внедрения правил безопасности пользователи должны быть ознакомлены с требованиями ИБ, а персонал пройти соответствующее обучение. При возникновении инцидентов работа должна вестись в соответствии с разработанными планами.

1.3.1. Шаблоны политик безопасности

Политика безопасности организации — это документ, описывающий специфические требования или правила, которые должны выполняться. В области информационной и сетевой безопасности политики обычно специфичны к области применения. *Стандарт* — это коллекция системно-специфических или процедурно-специфических требований, которые должен выполнять каждый пользователь. Ведущие организации, занимающиеся вопросами обеспечения информационной безопасности, разработали шаблоны ПБ. Например, институт SANS (System Administration, Networking, and Security) разработал серию шаблонов различных ПБ (www.sans.org/resources/policies/).

В число этих шаблонов входят, например, следующие политики:

- допустимая политика шифрования — определяет требования к криптографическим алгоритмам, используемым в организации;
- допустимая политика использования — определяет использование оборудования и компьютерных служб для защиты пользователей, ресурсов организации и собственно информации;
- антивирусная защита — определяет основные принципы эффективного уменьшения угрозы компьютерных вирусов для сети организации;
- политика оценки приобретений — определяет возможности покупок средств защиты организацией и определяет минимальные требования к оценке покупок, выполняемых группой информационной безопасности;
- политика аудита сканирования уязвимостей — определяет требования и назначает ответственного для сопровождения аудита и оценки риска, чтобы удостовериться в целостности информационных ресурсов, исследовать инциденты, устанавливать соответствие политикам безопасности или проводить мониторинг пользовательской и системной активности;
- политика автоматически передаваемой почты — документирует требования того, что никакая почта не может быть автоматически перенаправлена внешнему источнику без соответствующей санкции менеджера или директора;

• политика кодирования полномочий к базе данных (БД) — определяет требования для безопасного хранения и извлечения имен пользователей и паролей к БД;

• политика доступа по телефонной линии — определяет соответствующий доступ и его использование авторизованными персонами;

• политика безопасности демилитаризованной зоны — определяет стандарты для всех сетей и оборудования, применяемых в лаборатории, расположенной в демилитаризованной зоне или во внешних сетевых сегментах;

• политика критической информации — определяет требования к классификации и безопасности информации организации путем присвоения соответствующих уровней конфиденциальности;

• политика защиты паролей — определяет стандарты создания, защиты и смены паролей;

• политика удаленного доступа — определяет стандарты соединения с сетью организации из любого хоста или сети, являющихся внешними для организации;

• политика оценки риска — определяет требования и назначает ответственного для идентификации, оценки и уменьшения риска информационной инфраструктуры организации, ассоциированной с сопровождением бизнеса;

• политика безопасности маршрутизатора — определяет стандарты конфигурации минимальной безопасности для маршрутизаторов и коммутаторов внутри сети организации или используемых для работы (изготовления продукции);

• политика безопасности сервера — определяет стандарты конфигурации минимальной безопасности для серверов внутри сети организации или используемых как продукция;

• политика безопасности VPN — определяет требования для удаленного доступа IPSec или L2TP VPN соединений с сетью организации;

• политика беспроводных соединений — определяет стандарты для беспроводных систем, используемых для соединения с сетью организации.

Поэтому исходя из специфики построения и функционирования организации формируется соответствующий набор политик безопасности организации. Для обеспечения безопасности межсетевого взаимодействия особую роль играет сетевая политика безопасности.

1.3.2. Сетевая политика безопасности

При задании сетевой ПБ необходимо определить процедуры защиты своей сети, ее содержимого и пользователей от ущерба и потерь. С этой точки зрения сетевая ПБ играет роль проведения в жизнь

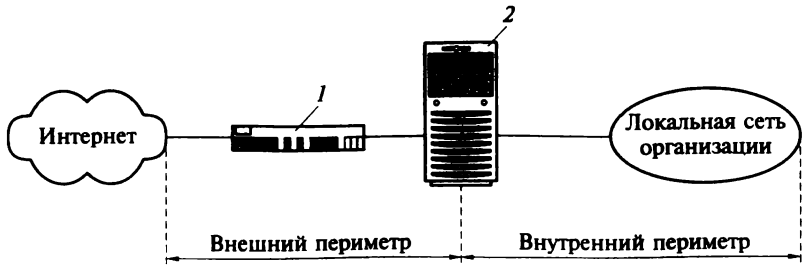


Рис. 1.6. Сетевой периметр организации:
1 — маршрутизатор; *2* — межсетевой экран

общей ПБ, определенной в организации. Сетевая ПБ концентрируется на контроле сетевого трафика и его использования. Она определяет сетевые ресурсы и угрозы, использование и возможности сети, а также детальные планы действий при нарушении ПБ.

При применении сетевой ПБ необходимо стратегически реализовать защитные границы внутри своей сети. Эти стратегические границы называются сетевыми периметрами. Устанавливая уровни безопасности сетевых периметров, можно осуществлять множественный контроль безопасности сетевого трафика.

Для установления сетевых периметров необходимо определить защищаемые компьютеры и сети, а также определить защитные механизмы для них. Одним из основных таких элементов является межсетевой экран. Чтобы установить успешный периметр безопасности, межсетевой экран (МЭ) должен быть шлюзом для всех соединений между доверенными сетями, не доверенными и неизвестными. Каждая сеть может содержать множество внутренних периметров. Чтобы понять, как сетевые периметры располагаются относительно друг друга, рассмотрим два типа сетевых периметров: внешний и внутренний.

В простейшем случае сетевой периметр организации включает в себя внешний и внутренний периметры (рис. 1.6).

Внешний сетевой периметр идентифицирует точку разделения между устройствами, которые контролируются, и теми, которые не контролируются. Обычно этой точкой является маршрутизатор, который используется для разделения своей сети от сети провайдера Интернета.

Внутренний сетевой периметр представляет собой дополнительные границы, в которых размещаются другие механизмы безопасности, такие как МЭ и фильтрующие маршрутизаторы. В этом случае внешний и внутренний периметры определены расположением внешнего и внутреннего маршрутизаторов и МЭ. Расположение МЭ между внутренним и внешним маршрутизаторами дает наибольшую дополнительную защиту от атак с обеих сторон, но значительно

уменьшает трафик, который должен исследовать МЭ, так как ему не нужно просматривать пакеты, циркулирующие во внутренней сети.

С точки зрения пользователей внешних сетей МЭ представляет собой все доступные компьютеры доверенной сети. Он определяет центральную точку, через которую должны проходить все соединения между двумя сетями.

Внешний сетевой периметр является наиболее небезопасной областью сетевой инфраструктуры организации. Обычно эта область предназначается для маршрутизаторов, МЭ и общедоступных интернет-серверов, таких как HTTP, FTP или e-mail. Поскольку к этой области легко получить доступ, она является наиболее часто атакуемой (обычно для того, чтобы через нее получить доступ к внутренней сети). Поэтому критичные данные, предназначенные только для внутреннего использования, не должны располагаться во внешнем сетевом периметре.

Можно использовать множество внутренних МЭ для установки множества внутренних сетевых периметров (рис. 1.7).

Использование внутренних межсетевых экранов позволяет ограничить доступ к совместно используемым внутренним ресурсам сети.

Доверенными сетями являются сети внутри сетевого периметра безопасности. Под *доверенными сетями* понимаются сети, над которыми специалисты организации имеют полный административный контроль. При установке МЭ необходимо точно идентифицировать типы сетей, которые присоединяются к МЭ посредством сетевых адаптеров. После начального конфигурирования доверенные сети включают МЭ и все сети позади него.

Недоверенными сетями являются сети, которые находятся вне установленного сетевого периметра. Они являются недоверенными, так как они вне контроля. При установке МЭ необходимо также точ-

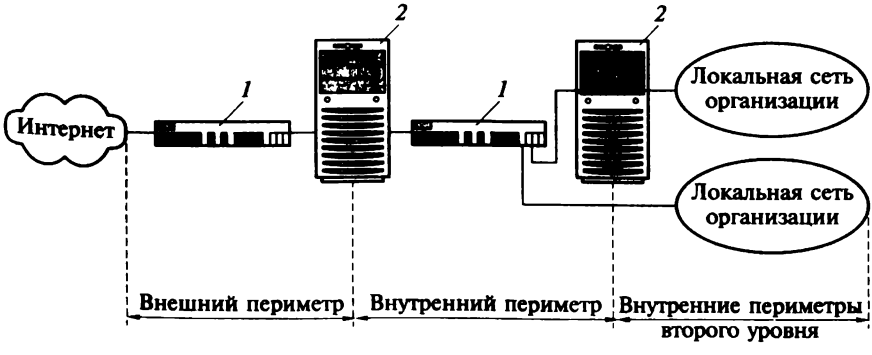


Рис. 1.7. Множество внутренних сетевых периметров:
1 — маршрутизатор; *2* — межсетевой экран

но определить недоверенные сети, от которых МЭ может допускать запросы.

Неизвестными сетями являются сети, которые не являются ни доверенными, ни недоверенными. Неизвестные сети существуют вне периметра безопасности (по умолчанию все недоверенные сети рассматриваются как неизвестные).

Область внешнего сетевого периметра называют демилитаризованной зоной (Demilitarized Zone, DMZ). *Демилитаризованная зона* — по международному праву территория, на которой ликвидированы военные укрепления и сооружения, запрещено содержание вооруженных сил.

В русском языке есть более подходящее название — «нейтральная зона» (территория, географический район, где по международному соглашению или решению государства, к которому он принадлежит, запрещается размещение военных объектов, подготовка и ведение боевых действий). Так как аббревиатура DMZ прижилась в Интернете и компьютерных публикациях, будем использовать ее.

Важной составной частью политики безопасности является сетевая политика безопасности или политика сетевого подключения. Политикой сетевого подключения должны быть определены типы устройств, разрешенные для подключения к сети. В ней должны быть детально определены настройки систем, которые допускается подключать к сети. Эта политика может включать в себя следующие разделы:

- описание процесса установки и настройки операционной системы (ОС) и приложений, а также их функциональных возможностей, которые разрешено использовать;
- местоположение в сети (физической подсети) систем определенного типа и процедура разрешения вопросов адресации в сети;
- требование об установке и регулярном обновлении антивирусного ПО;
- описание настройки прав пользователей и защиты ресурсов, обеспечиваемых ОС;
- процедуры, которым необходимо следовать для создания новой учетной записи пользователя, и аналогичные процедуры для ее удаления;
- запрет на установку дополнительных аппаратных или программных компонентов без одобрения сетевого администратора (или другого ответственного лица).

1.3.3. Эшелонированная оборона

Под *эшелонированной обороной* (defense in depth) в современной компьютерной литературе понимается практическая стратегия достижения информационной гарантированности (information

assurance) в сетевом оборудовании. Эта стратегия представляет собой баланс между свойствами защиты и стоимостью, производительностью и функциональными характеристиками.

Информационная гарантированность достигается, когда информация и информационные системы защищены от атак посредством применения служб безопасности, таких как доступность, целостность, аутентификация, конфиденциальность и неотказуемость. Приложения, реализующие эти службы, должны базироваться на парадигме — защита, обнаружение и реакция.

Достижение информационной гарантированности стратегии эшелонированной обороны требует нахождения баланса трех основных элементов.

1. Персонал — достижение информационной гарантированности начинается с уровня управления организацией. Персонал управления должен ясно представлять себе возможные угрозы выполнению целевых задач организации. За этим должны следовать инвентаризация ресурсов, определение политики и процедур обеспечения информационной гарантированности, распределение ролей персонала и определение ответственности. Сюда же относятся процедуры физической защиты и меры безопасности персонала.

2. Технология — для того чтобы увериться, что достаточные технологии выбраны и правильно применены, необходимы разработка и внедрение эффективной политики и процедур обеспечения информационной гарантированности. Они должны включать в себя: политику безопасности, архитектуру и стандарты системного уровня, критерии выбора необходимых продуктов, специфику конфигурирования компонентов систем, а также процедуры оценки рисков.

3. Функциональные операции — данный аспект фокусируется на всей ежедневной деятельности, требуемой для поддержания безопасности организации. В состав таких функциональных операций могут входить, например, следующие операции: разработка, установка и поддержание политики безопасности (политик безопасности); проверка и сертификация изменений в используемых информационных технологиях; управление установленными средствами обеспечения безопасности; контроль и реагирование на текущие угрозы; обнаружение атак и реакция на них; процедуры восстановления и переустановки компонент информационных технологий и т. д.

Стратегия эшелонированной обороны рекомендует применение следующих принципов информационной гарантированности для технологии:

• применение защиты во множественных местах. Поскольку злоумышленники могут атаковать систему из множества мест, включая внешние и внутренние, организация должна применять защитные механизмы в различных точках, которые должны обеспечивать защиту сетей и инфраструктуры, защиту границ сети и территории, а также защиту компьютерного оборудования;

• применение уровневой защиты предполагает установку защитных механизмов между потенциальным злоумышленником и целью;

• определение устойчивости безопасности достигается оценкой защитных возможностей каждого компонента информационной гарантированности;

• применение инфраструктуры обнаружения атак и вторжений, использование методов и средств анализа и корреляции получаемых данной инфраструктурой результатов.

Концепция эшелонированной обороны в настоящее время является общепринятой, поэтому производители средств защиты реализуют ее выпуском целых линеек защитных средств, которые функционируют совместно и управляются, как правило, единым устройством управления.

1.4. Управление рисками

Одной из важнейших задач управления ИБ является управление рисками. Управление риском охватывает три процесса: оценку риска, уменьшение риска, применение результатов оценки. Управление риском есть процесс, который позволяет менеджерам информационных технологий (ИТ) осуществлять баланс между операционной и экономической стоимостями защитных мер и достигать целей защиты ИТ-систем и данных, которые поддерживают их организационную миссию. Поэтому управление риском — одна из главных задач и обязанностей управления организации.

Управление рисками ИБ — итеративный процесс, который требует контроля и периодического пересмотра.

1.4.1. Основные понятия

Управление рисками использует понятийный аппарат, который в настоящее время стандартизован.

Ресурсы (assets) — это то, что организация ценит и хочет защитить. Ресурсы включают в себя информацию и поддерживающие средства, которые требуются организации для ведения бизнеса. Примерами ресурсов являются:

• информация (данные), т.е. файлы с деталями платежей, звуковые записи, файлы изображений, информация о продуктах, описания и планы;

• бумажные документы (контракты, заполненные формы);

• программное обеспечение (системное ПО, прикладное ПО, средства разработки и утилиты);

• физическое оборудование (компьютеры и коммуникационное оборудование, магнитные носители, другое техническое оборудование);

• службы (вычислительные и коммуникационные, провайдеры служб, утилиты);

• люди и их знания (техническое, операционное, маркетинговое, юридическое, финансовое);

• имидж и репутация организации.

Каждому ресурсу должно быть присвоено значение. Значения ресурсов используются для идентификации соответствующей защиты и определения важности ресурсов для бизнеса. Значения могут быть выражены в терминах потенциального влияния на бизнес нежелательных событий, приводящих к потере конфиденциальности, целостности и (или) доступности. Потенциальное влияние может включать в себя финансовые потери, падение доходов и акций на рынке или падение престижа.

Угроза (threat) — потенциальная причина нежелательного события, которое может нанести ущерб организации и ее объектам. Угрозы могут быть естественными (наводнение, пожар, землетрясение), преднамеренными или случайными. Результатом реализации угрозы может быть:

• разрушение объекта (средств, данных, оборудования, связи);

• повреждение или модификация объекта (данных, приложений);

• кража, удаление или потеря объекта (оборудования, данных, приложений);

• раскрытие объекта (данных);

• использование или внедрение нелегального объекта (оборудования, нелицензированного ПО, фальшивых данных);

• прерывание службы.

Угроза требует реализации уязвимости объекта, чтобы вызвать ущерб.

Уязвимости (vulnerability) — слабости, ассоциированные с ресурсами организации. Уязвимость есть условие или множество условий, которые могут позволить угрозе воздействовать на объект или с большей частотой, или с большим влиянием, или совместно.

Обычно уязвимость является следствием плохо отработанных процедур, ошибок непрофессионального персонала, некорректной конфигурации или исследовательской технологии. Чтобы уязвимость была использована, она должна быть известна или реализована угрозой.

Риск безопасности — возможность данной угрозы реализовать уязвимости для того, чтобы вызвать ущерб или разрушение ресурса или группы ресурсов, что прямо или косвенно воздействует на организацию. Уровень риска безопасности определяется комбинацией значений ресурсов, оцененных уровней соответствующих угроз и ассоциированных с ними уязвимостей.

Контроль безопасности — практика, процедуры и механизмы, которые могут защитить объекты от угроз, уменьшить уязвимости или уменьшить влияние нежелательных событий.

1.4.2. Процесс оценки рисков

Оценка риска — основной процесс в методологии управления риском. Риск является функцией вероятности того, что данный источник угроз осуществит частную потенциальную уязвимость, и результирующего влияния нежелательного события на организацию.

Взаимоотношения между компонентами оценки риска представлены на рис. 1.8.

Для определения вероятности будущих нежелательных событий угрозы для ИТ-системы должны быть рассмотрены вместе с потенциальными уязвимостями и привязаны к местоположению в ИТ-системе. Влияние связано с величиной ущерба, который может быть вызван угрозой, реализующей уязвимость. Уровень влияния определяется потенциальным предназначением влияния, и ему присваивается относительное значение для ИТ-ресурса и ресурсов, на которые оно воздействует (т. е. критичность и чувствительность ИТ-компонентов и данных).

Общая оценка риска включает в себя следующие шаги:

1) характеристика системы;
2) идентификация угроз;
3) идентификация уязвимостей;
4) анализ средств защиты (контроля);
5) определение вероятностей (ранжирование частот появления);

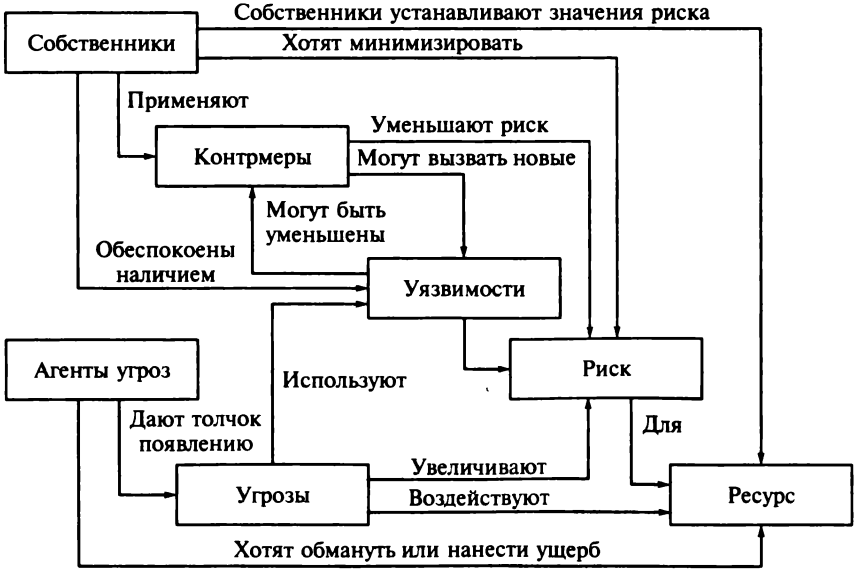


Рис. 1.8. Взаимоотношения между компонентами оценки риска

6) анализ влияния;
7) определение риска;
8) рекомендации по средствам защиты (контролю);
9) результирующая документация.

Характеристика системы. Она включает в себя описание системы, системных компонентов, элементов системы, пользователей и т. п. В нее также включают расположение частей (подразделений) организации, основные информационные потоки, классификацию информации и другую необходимую информацию, которая может использоваться на следующих шагах.

Идентификация угроз. Источник угроз — это или намеренное и методическое исследование цели в поисках уязвимостей, или ситуация и метод, которые случайно приводят к уязвимости.

Источник угроз определяется как любое обстоятельство или событие, которые потенциально могут нанести ущерб ИТ-системе (табл. 1.2).

Таблица 1.2. **Источники и описания угроз**

Источник угроз	Мотивация	Действие угрозы
Хакеры	Вызов Бунтарство Самооценка	Взлом Социальная инженерия Неавторизованный доступ
Криминальные элементы	Уничтожение информации Денежная выгода Хищение данных	Враждебные действия Системное вторжение Просмотр данных
Террористы	Уничтожение данных Получение выгоды Использование ресурсов	Информационная война Отказ в обслуживании Проникновение в систему
Промышленный шпионаж	Экономический шпионаж Конкурентная борьба	Кража информации Внедрение в систему Неавторизованный доступ
Пользователи (внутренние нарушители)	Случайные ошибки Денежная выгода Обида	Просмотр данных Неавторизованный доступ

Для определения системных уязвимостей рекомендуют использование источников угроз, результатов тестирования системной безопасности, разработку контрольного списка требований по безопасности.

Типы уязвимостей и методологии их определения обычно зависят от природы ИТ-системы и фазы ее жизненного цикла:

- если ИТ-система еще не спроектирована, то поиск уязвимостей концентрируется на организационной ПБ, планируемых процедурах безопасности, определении системных требований и анализе документов поставщиков продуктов безопасности;
- если ИТ-система уже применяется, то идентификация должна быть расширена для включения дополнительной информации, такой, например, как соответствие свойств безопасности, описанных в документации по безопасности, результатам сертификационных тестов и испытаний;
- если ИТ-система функционирует, то процесс идентификации уязвимостей должен включать в себя анализ свойств безопасности ИТ-системы и контроля безопасности (технического и процедурного), используемого для защиты системы.

Идентификация уязвимостей. Технические и нетехнические уязвимости, ассоциированные с обрабатывающим оборудованием ИТ-системы, могут быть идентифицированы посредством применения технологий сбора информации. Другим существенным источником данных по уязвимостям является Интернет. Часто в сети разработчиками публикуются известные системные уязвимости вместе со средствами их устранения.

Могут применяться и другие документированные источники, например:

- результаты предыдущих оценок риска;
- отчеты аудита ИТ-системы, отчеты о системных аномалиях, отчеты с обзорами безопасности, отчеты о системных тестах и оценках;
- списки уязвимостей, например, из баз данных уязвимостей;
- информационные бюллетени по безопасности;
- информационные бюллетени изготовителей;
- коммерческие компании, выполняющие функции информационного аудита безопасности;
- анализ безопасности системного ПО;
- тестирование системной безопасности.

Для оценки реального наличия уязвимостей могут применяться методы тестирования (включая системное тестирование) для идентификации системных уязвимостей в зависимости от критичности ИТ-системы и доступных ресурсов (т.е. выделенного бюджета, доступной технологии, наличия квалифицированного персонала для осуществления тестирования). Методы тестирования включают в себя:

- средства автоматического сканирования уязвимостей;

• тесты и оценка безопасности;
• тесты на проникновение.

Средства автоматического сканирования уязвимостей используются для сканирования групп хостов или сети на известные уязвимые службы. Необходимо заметить, что некоторые потенциальные уязвимости, определенные таким автоматическим средством, могут не представлять реальных уязвимостей в контексте реального системного оборудования организации.

Тестирование и оценка безопасности могут быть использованы для идентификации уязвимостей во время процесса оценки риска. Они включают в себя разработку и выполнение плана тестирования (т.е. разработку программных средств тестов, тестовые процедуры, ожидаемые результаты тестов). Назначение тестирования — протестировать эффективность контроля безопасности в ИТ-системе, т.е. как он применен в операционном оборудовании. Цель — удостовериться, что прикладной контроль удовлетворяет спецификациям безопасности для ПО и аппаратуры. Кроме того, оценивается то, как прикладной контроль согласовывается с существующими требованиями (например, стандартами).

Тесты на проникновение могут использоваться для оценки контроля безопасности и уверенности в том, что различные аспекты ИТ-системы защищены. Тесты на проникновение могут использоваться для оценки способности ИТ-системы противостоять попыткам нарушения системной безопасности. Их целью является тестирование ИТ-системы с точки зрения третьей стороны для идентификации потенциальных отказов защитных систем ИТ-системы.

На основе анализа уязвимостей составляется список системных уязвимостей, которые могут быть вызваны потенциальными источниками угроз.

Анализ средств защиты (контроля). Целью этого шага является анализ применяемых или планируемых к применению средств защиты (контроля) в организации для минимизации или устранения вероятности уязвимости, реализуемой источником угроз.

Определение вероятностей (ранжирование частот появления). Для получения общей оценки вероятности, которая показывает вероятность того, что потенциальная уязвимость может быть реализована внутри конструкции ассоциированного с угрозой оборудования, могут быть рассмотрены следующие факторы:
• движущая сила (мотивация) и способность источника угроз;
• природа уязвимости;
• существование и эффективность текущего контроля.

Поскольку получение количественных данных вероятностей затруднено, обычно используются нечеткие шкалы. Например, вероятность того, что потенциальная уязвимость может быть реализована данным источником угроз, может быть описана как высокая, средняя и низкая (табл. 1.3).

Таблица 1.3. **Значения уровней вероятностей**

Уровень частоты	Описание частоты
Высокий	Источник угроз имеет высокую мотивацию и значительную способность, существующий контроль для защиты от реализации неэффективен
Средний	Источник угроз мотивирован и имеет способность к реализации уязвимостей, но имеющийся контроль осложняет реализацию уязвимости
Низкий	Источник угроз имеет недостаточную мотивацию и способность, контроль защищает или значительно осложняет реализацию уязвимости

Выходом данного шага являются ранжированные оценки частот появления.

Анализ влияния. При проведении анализа влияния цель состоит в определении нежелательного влияния успешной реализации уязвимостей угрозами. Для этого необходимо получить следующую информацию:

- миссия системы (т.е. процессы, осуществляемые ИТ-системой);
- критичность системы и данных (т.е. значение или важность системы для организации);
- чувствительность системы и данных.

Степень влияния также оценивается нечеткой шкалой (табл. 1.4).

Определение риска. Назначение этого шага — оценить уровень риска ИТ-системы. Определение риска для частной пары угроза-уязвимость может быть выражена как функция:

- от вероятности того, что данный источник угроз пытается реализовать данную уязвимость;
- величины влияния при удачной реализации источником угроз уязвимости;
- достаточности планируемого или существующего контроля для уменьшения или устранения риска.

Для измерения риска используются шкалы риска и матрица уровней риска. В матрице уровней риска окончательное определение задачи риска получается умножением уровней частоты (классов) вероятностей угроз на степень влияния угроз. Такая матрица показывает, как общие классы риска могут быть определены на основе входов из категорий вероятности угрозы и влияния угрозы (табл. 1.5).

Данная матрица (3×3) является матрицей оценок угроз (высокой, средней, низкой) и влияния угрозы (высокое, среднее, низкое). В за-

Таблица 1.4. **Оценка степени влияния**

Величина влияния	Описание влияния
Высокая	Реализация уязвимости может вызвать дорогостоящие потери материальных ресурсов, нанести вред, значительно подорвать репутацию или вызвать человеческие травмы
Средняя	Реализация уязвимости может вызвать некоторую потерю материальных ресурсов, нанести вред, подорвать репутациюс или вызвать человеческие травмы
Низкая	Реализация уязвимости может вызвать потери некоторых материальных ресурсов или заметно воздействовать на репутацию и миссию организации

висимости от требований организации и числа градаций желаемой оценки риска можно использовать матрицы 4×4 или 5×5. Например, в матрицу 5×5 могут быть добавлены следующие значения:

- для вероятности угрозы — очень низкая, очень высокая;
- для влияния угрозы — очень низкая, очень высокая.

Это позволит получить очень низкий и очень высокий уровни риска. Очень высокий уровень риска может требовать выключения системы или остановки всех средств интеграции ИТ-системы.

Пример, приведенный в табл. 1.5, показывает, как получаются общие уровни риска. Определение этих уровней риска или классов может быть в значительной степени субъективным. Логическое обо-

Таблица 1.5. **Матрица уровней (классов) рисков**

Вероятность угрозы	Влияние		
	Низкое (10)	Среднее (50)	Высокое (100)
Высокая (1,0)	Низкий 10×1,0 = 10	Средний 50×1,0 = 50	Высокий 100×1,0 = 100
Средняя (0,5)	Низкий 10×0,5 = 5	Средний 10×0,5 = 25	Высокий 100×0,5 = 50
Низкая (0,1)	Низкий 10×0,1 = 1	Средний 50×0,1 = 5	Высокий 100×0,1 = 10

снование может быть проведено в терминах вероятностей, присвоенных каждому уровню вероятности угрозы, и значения, присвоенного каждому уровню влияния. Например, присвоить вероятность для уровня угрозы высокий — 1,0; для среднего — 0,5; для низкого — 0,1. Для значений влияния: для высокого — 100; для среднего — 50; для низкого — 10.

Рекомендации по средствам защиты (контроля). На основе матрицы уровней рисков выбираются средства защиты (контроля), которые могут уменьшить или устранить идентифицированный риск. Целью этих рекомендаций является уменьшение уровня риска ИТ-системе и ее данным до допустимого уровня. Рекомендации по средствам защиты (контроля) являются результатом процесса оценки риска и дают исходные данные для процесса уменьшения риска, во время которого оценивается рекомендуемый процедурный и технический контроль безопасности, назначаются приоритеты и выполняются планы покупки и внедрения различных средств.

Необходимо отметить, что не все рекомендуемые меры контроля могут быть применены для уменьшения потерь. Для определения конкретных мер, которые требуются и соответствуют определенной организации, применяется анализ затрат и результатов (cost/benefit analysis). Данный анализ применяется к рекомендуемым средствам защиты, чтобы продемонстрировать, что стоимость применения контроля может быть оправдана снижением уровня риска. Дополнительно должны быть учтены такие влияния рекомендуемых опций, как операционное влияние (т. е. влияние на производительность системы) и осуществимость (т. е. выполнение технических требований, пользовательская приемлемость).

В результате выполнения этого шага разрабатываются рекомендации по средствам защиты (контроля) и альтернативные решения для уменьшения риска.

Результирующая документация. После выполнения оценки риска (источники угроз и уязвимости идентифицированы, риск оценен, рекомендации по контролю подготовлены) результаты должны быть представлены в виде официального отчета. Соответствующий отчет по оценке риска описывает угрозы, уязвимости, измерения риска и дает рекомендации по применению средств защиты (контроля).

1.4.3. Уменьшение рисков

После получения результатов анализа рисков основной задачей является разделение рисков на приемлемые и неприемлемые. Если риск является неприемлемым, то необходимо применять процедуры его уменьшения. Процесс уменьшения риска включает в себя установление приоритетов отдельным составляющим общей оценки

риска, оценку и применение соответствующих уменьшающих риск рекомендованных средств защиты (контроля). Поскольку устранение всех рисков обычно невозможно или непрактично, обязанностью администрации является выполнение анализа затрат и результатов и применение наиболее соответствующих средств защиты для уменьшения риска до допустимого уровня, характеризующегося минимальным нежелательным влиянием на ресурсы и миссию организации.

В общем случае уменьшение риска может быть достигнуто посредством применения одной или комбинации следующих операций уменьшения риска:

- принятие риска (принять потенциальный риск и продолжать функционирование ИТ-системы или применить средства защиты для уменьшения риска до допустимого уровня);
- уклонение от риска (уклониться от риска, устранив причину риска и (или) последствий, т.е. отказаться от определенных функций системы или выключать систему, когда идентифицируется риск);
- ограничение риска (ограничить риск применением средств защиты, которые минимизируют нежелательное влияние реализации уязвимости угрозой, т.е. использовать поддерживающие, предупредительные или обнаруживающие средства);
- планирование риска (управлять риском, разрабатывая план уменьшения риска, который устанавливает приоритеты, применяет и поддерживает средства защиты);
- исследования и подтверждения (уменьшить риск потерь подтверждением уязвимости или дефекта и применением исследовательских средств защиты для устранения уязвимости);
- передача риска (передать риск, используя другие варианты для компенсации потерь, такие, например, как покупка страховки).

При принятии решения для уменьшения риска должны быть рассмотрены цели и миссия организации. Может оказаться непрактичным рассматривать все идентифицированные риски, поэтому должны быть установлены приоритеты для пар угроза—уязвимость, которые имеют потенциальную возможность оказывать наибольшее влияние на миссию или ущерб. Кроме того, при защите миссии организации и ее ИТ-систем (поскольку каждая организация имеет уникальное оборудование и решает свои задачи) варианты, используемые для уменьшения риска, и методы, используемые для применения средств защиты, могут варьироваться. Наилучшим подходом является использование соответствующих технологий, в число которых входят использование средств защиты различных производителей, соответствующие варианты уменьшения риска и нетехнические административные меры.

Общая схема процесса уменьшения риска приведена на рис. 1.9. Соответствующие точки применения действий средств защиты на рисунке помечены словом «да».

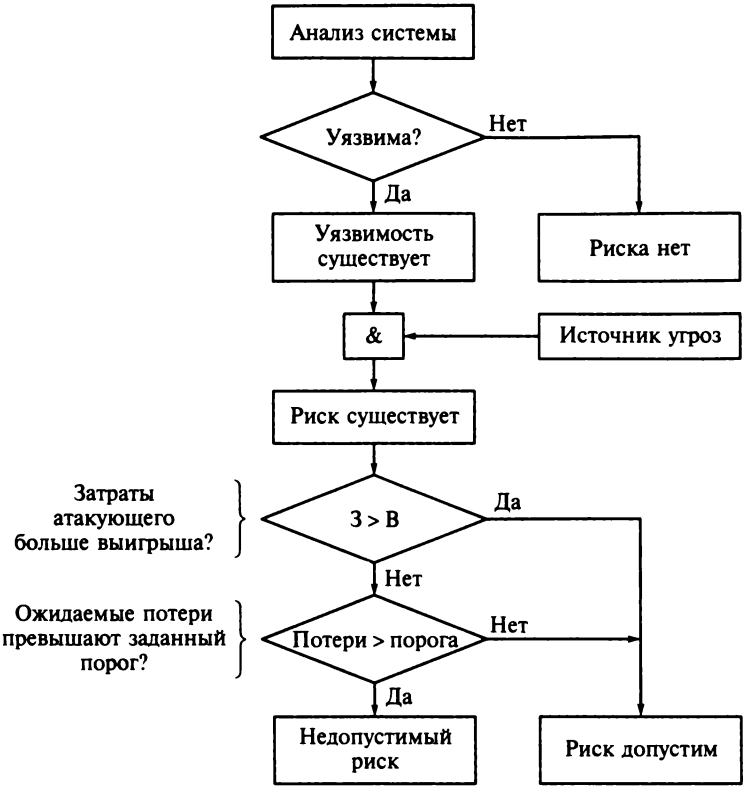


Рис. 1.9. Схема процесса уменьшения риска

Рассмотрим правила, которые являются руководством по действиям уменьшения риска от преднамеренных угроз:

1) когда уязвимость (или дефект, слабость) существует — применить технику передачи (assurance) для уменьшения вероятности реализации уязвимости;

2) когда уязвимость может быть осуществлена — применить уровневую защиту, специальное проектирование архитектуры и административные средства для минимизации риска или предупреждения его;

3) когда затраты атакующего меньше, чем потенциальный выигрыш, — применить защиту для уменьшения мотивации атакующего так, чтобы увеличить затраты атакующего (т.е. использовать такие средства, которые ограничивают то, к чему может иметь доступ пользователь, и значительно уменьшают выигрыш атакующего);

4) когда потери (убытки) очень велики — применить принципы конструирования, архитектуру, техническую и нетехническую защи-

ту для ограничения степени влияния атаки и, таким образом, уменьшить потенциал потерь.

Такая стратегия, за исключением третьего пункта, также применима для уменьшения риска, возрастающего в результате аппаратных или непреднамеренных человеческих угроз (т.е. ошибок системы или пользователей).

Как правило, основным методом снижения риска является применение соответствующих средств защиты (контроля). При этом необходимо применять следующее правило: «В первую очередь рассматривать наибольший риск, стремясь к значительному уменьшению риска при минимальной стоимости и минимальном влиянии на другие свойства рассматриваемого объекта».

Для уменьшения риска используется следующая методология, описывающая подход к применению средств защиты.

Шаг 1 — присвоить приоритеты действиям. Базируясь на уровнях риска, представленных в отчете по оценке риска, расставить приоритеты действиям. При применении к ресурсам наивысший приоритет должен быть дан значениям риска с неприемлемо большими рангами (т.е. риску присвоен очень высокий или высокий уровень). Эти пары угроза—уязвимость будут требовать немедленных корректирующих действий для защиты интересов и миссии организации. Выход шага 1 — ранжированные (от высокого до низкого) риски.

Шаг 2 — оценить рекомендуемые варианты защиты. Рекомендованные в результате процесса оценки риска средства защиты могут быть неподходящими и неосуществимыми для определенной организации и ИТ-системы. Во время этого шага анализируется их осуществимость (т.е. совместимость с существующими средствами и системами, возможность принятия пользователями) и эффективность (т.е. степень защиты и уровень уменьшения риска) рекомендованных средств защиты. Цель — выбрать наиболее подходящие средства для минимизации риска. Выход шага 2 — список подходящих средств защиты (контроля).

Шаг 3 — провести анализ затрат и результатов. Анализ выполняется для помощи руководству в принятии решения и идентификации эффективной стоимости средств. Выход шага 3 — анализ затрат и результатов, описывающий стоимость и результаты применения (или не применения) средств защиты.

Шаг 4 — выбрать средства защиты. На основе анализа затрат и результатов управление организации определяет средства защиты для уменьшения риска для миссии организации на основе критерия стоимость—эффективность. Выбранные средства должны комбинировать технические, операционные и управляющие средствами элементы, чтобы обеспечить адекватную защиту ИТ-системы и организации. Выход шага 4 — список выбранных средств.

Шаг 5 — назначить ответственных. Идентифицируются подходящие персоны (из числа внутреннего персонала или с привлечением внешних организаций), которые имеют соответствующий опыт и

знания в применении выбранных средств защиты, и назначаются ответственными. Выход шага 5 — список ответственных персон.

Шаг 6 — разработать план реализации защиты. На этом шаге разрабатывается план реализации (применения) средств защиты (план действий). Этот план, как минимум, должен содержать следующую информацию:

- риски (пары угроза—уязвимость) и соответствующие уровни риска (выход отчета оценки риска);
- рекомендуемые средства защиты (из отчета оценки риска);
- приоритеты действий (с приоритетами, данными с учетом очень высокого и высокого уровней риска);
- выбранные и планируемые к применению средства защиты (определенные на основе осуществимости, эффективности и выигрыша для организации и стоимости);
- требуемые ресурсы для применения выбранных планируемых средств;
- списки ответственных команд и персонала;
- начальная дата применения;
- назначенная дата завершения применения;
- требуемые средства и необходимое обслуживание.

Этот план расставляет приоритеты действиям по применению средств защиты и составлен от даты начала до даты конца. Он поможет упростить процесс уменьшения риска. Выход шага 6 — план применения средств защиты.

Шаг 7 — применить выбранные средства защиты. В зависимости от индивидуальной ситуации применение средств защиты может уменьшить уровень риска, но не уничтожить его. Выход шага 7 — остаточный риск.

При применении рекомендованных средств для уменьшения риска организация должна рассмотреть технические, управленческие и

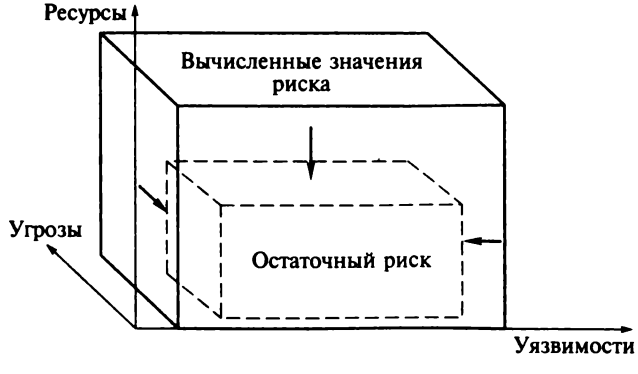


Рис. 1.10. Схема получения остаточного риска

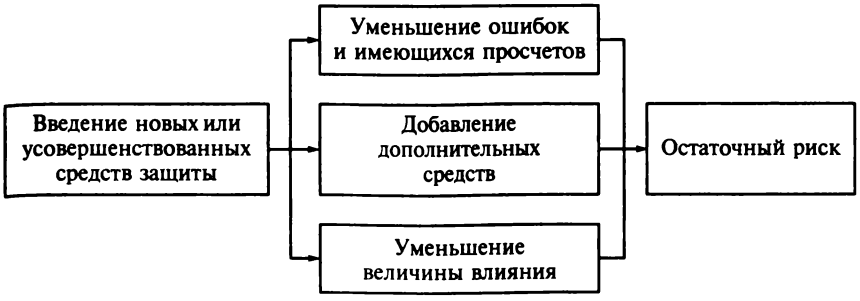


Рис. 1.11. Процедуры уменьшения уровня риска

операционные средства защиты или их комбинацию, чтобы обеспечить максимальную эффективность защиты для своей ИТ-системы и организации. Средства защиты, используемые надлежащим образом, могут защитить, ограничить или остановить ущерб источника угроз.

Остаточный риск. Применение средств защиты не гарантирует уничтожение риска, более того, это в принципе невозможно. Поэтому всегда остается какой-то уровень риска, называемый *остаточным риском*. Цель процесса уменьшения риска и состоит в получении остаточного риска, который допустим для выполнения миссии организации. Общая схема получения остаточного риска показана на рис. 1.10.

Организация может анализировать оценку уменьшения риска в результате применения новых или усовершенствованных средств защиты в терминах уменьшения вероятности угрозы или влияния — двух параметров, которые определяют уменьшение уровня риска миссии организации (рис. 1.11).

Для получения допустимого остаточного риска организация может использовать новые или усовершенствованные средства защиты, которые уменьшают риск посредством:

- удаления некоторых системных уязвимостей (дефектов и слабостей) путем уменьшения числа возможных пар источник угрозы—уязвимость;
- добавления специальных нацеленных средств для уменьшения способности и мотивации источника угроз. Например, если определено, что стоимость инсталляции и поддержки дополнительного ПО безопасности для отдельного компьютера, хранящего чувствительную информацию, слишком высока для организации, то может быть применен усиленный административный и физический контроль, чтобы затруднить физический доступ к данному компьютеру;
- уменьшения величины нежелательного влияния (например, ограничение степени уязвимости или модификация взаимоотношений между ИТ-системой и миссией организации).

Вычисление и оценивание риска. Тенденции развития организаций показывают, что их компьютерные сети непрерывно расширяются и обновляются, компоненты сетей и ИТ меняются и дополняются, прикладное ПО заменяется или обновляется новыми версиями. Кроме того, изменения затрагивают персонал организации и соответственно политику безопасности. Эти изменения означают, что появляются новые риски, а риски, ранее подвергшиеся уменьшению, снова становятся значительными. Поэтому процесс управления рисками должен быть непрерывным и эволюционирующим.

В США процесс оценки риска обычно повторяется каждые три года государственными органами (OMB Circular A-130). Управление риском должно интегрироваться с жизненным циклом ИТ-системы, потому что это является хорошей практикой и поддержкой выполнения бизнеса организацией. Периодическое проведение процесса оценки риска позволит гибко и быстро реагировать на изменения в ИТ-системе и обслуживающем оборудовании. Для успешного выполнения своей миссии организация должна иметь соответствующую программу управления риском.

1.4.4. Пример содержания результирующего отчета

Рассмотрим примерную структуру отчета оценки риска.

1. Введение:
   - цель;
   - масштабы данной оценки риска.

Описание системных компонент, элементов, пользователей, расположение частей организации и другие детали системы, включенные в рассмотрение для оценки риска.

2. Подход к оценке риска — кратко описывается подход, используемый для оценки риска:
   - участники (т. е. члены команды оценки риска);
   - техника, используемая для сбора информации;
   - разработка и описание шкалы рисков (т. е. 3×3, 4×4 или 5×5 матрицы уровней риска).

3. Характеристика системы — характеризуются система, включая аппаратуру (серверы, маршрутизаторы, коммутаторы), программное обеспечение (приложения, ОС, протоколы), системные интерфейсы (коммуникационные каналы), данные (хранимые, передаваемые и обрабатываемые), и пользователи системы. Приводятся диаграммы связности или схемы системных входов и выходов для очерчивания границ оценки риска.

4. Ведомость угроз — собираются и перечисляются потенциальные источники угроз и ассоциированные с ними действия, соответствующие оцениваемой системе.

5. Результаты оценки риска — перечисляются все рассматриваемые утверждения (пары угроза—уязвимость). Каждое утверждение должно включать в себя:

- номер утверждения и краткое описание (например, «Утверждение 1: пароли системных пользователей могут быть подобраны или взломаны»);
- обсуждение источников пар угроз—уязвимостей;
- идентификация существующих средств защиты;
- обсуждение частоты появления и оценок влияния (высокая, средняя, низкая);
- обсуждение анализа и оценки влияния (высокое, среднее, низкое влияние);
- ранжирование риска на основе матрицы уровней риска (высокий, средний, низкий уровень риска);
- рекомендуемые средства или альтернативные варианты для уменьшения риска.

6. Заключение — сводка утверждений, ассоциированных уровней риска, рекомендации и любые комментарии, сведенные в таблицу для удобства пользования руководством организации и облегчения применения рекомендованных средств во время процесса уменьшения риска.

1.5. Аудит информационной безопасности

Под *аудитом* информационной системы (информационной технологии) понимается системный процесс получения и оценки объективных данных о текущем состоянии системы (технологии), действиях и событиях, происходящих в ней, который устанавливает уровень их соответствия определенному критерию и предоставляет результаты заказчику.

Проведение аудита позволяет оценить текущую безопасность функционирования ИТ, оценить риски и управлять ими, прогнозировать их влияние на бизнес-процессы организации, корректно и обоснованно подходить к вопросу обеспечения безопасности информационных активов организации, основными из которых являются:

- идеи;
- знания;
- проекты;
- результаты внутренних обследований.

Общее понятие аудита. Датой рождения аудита принято считать 1844 г., когда в Англии приняли закон об акционерных компаниях, согласно которому их правления должны были ежегодно отчитываться перед аукционерами, причем отчет должен был быть проверен и подтвержден специальным человеком — независимым аудитором.

В настоящее время аудит, пройдя несколько этапов своего развития, стал частью хозяйственной жизни развитых стран. От проверки бухгалтерских счетов акционерных компаний отдельными профессиональными аудиторами аудит развился до комплексного понятия, включающего в себя ряд услуг (проверку бухгалтерской отчетности, финансовый анализ, консультирование), оказываемых профессиональными аудиторами и аудиторскими фирмами. Среди таких фирм есть и небольшие, включающие в себя десяток сотрудников, и гиганты с численностью до нескольких тысяч человек. Назовем компании из так называемой большой пятерки: Ernest & Young, CPMG, Price Waterhouse Coopers, Arthur Anderson, Deloyt & Touche.

Концепции регулирования аудиторской деятельности в мировой практике различны. В Германии, Франции и Испании, где основными пользователями бухгалтерской отчетности считаются государственные организации и банки, более распространено государственное регулирование аудиторской деятельности. В Великобритании и США, где основными пользователями бухгалтерской отчетности считаются аукционеры, кредиторы, инвесторы, более распространено саморегулирование.

В Российской Федерации система регулирования аудиторской деятельности, особенно в сфере ИТ, находится в процессе становления. В настоящее время преобладает государственное регулирование, но появляются и элементы саморегулирования. Уполномоченным органом, регулирующим аудиторскую деятельность в Российской Федерации, является Министерство финансов Российской Федерации (основным документом является Федеральный закон № 307-ФЗ от 30 декабря 2008 г. «Об аудиторской деятельности»). При этом при разработке общероссийских стандартов принимаются во внимание международные стандарты, которые призваны решать те же задачи.

Внутрифирменные стандарты решают те же задачи, что и международные и общероссийские, но в масштабе аудиторской фирмы. Каждая аудиторская организация должна сформировать пакет своих внутренних (внутрифирменных) стандартов, отражающих подход к проведению проверки и составлению заключения.

Виды аудита. Аудит безопасности информационных систем обычно подразделяют на внешний и внутренний.

Подробнее рассмотрим эти виды аудита применительно к сетевым технологиям, применяемым современными ИС.

Внешний аудит проводится в основном вне организации и, как правило, специализированными организациями, занимающимися аудитом информационной безопасности. В этом случае анализируются меры риска от внешних атак и атак со стороны (даже если организация защищена межсетевыми экранами). При проведении внешнего аудита осуществляется сканирование портов, поиск уязвимостей в сетевом и прикладном программном обеспечении.

Осуществляются попытки взаимодействия с Web-серверами, почтовым и файловыми серверами, попытки вхождения в локальные сети организации.

По желанию руководства организации может проводиться специальный вид внешнего аудита — Ethical Hacking. В этом случае специальная организация (в мире это является широко распространенной практикой, такие подразделения имеют специальное название tiger team) осуществляет избранные атаки на серверы, сайты и хосты организации. Такие атаки могут продемонстрировать уязвимости ИС организации.

Внутренний аудит, как правило, проводится специальной командой из числа персонала организации. Его задачей является оценка риска существующей технологии применения ИС. Этот вид аудита выполняется с привлечением средств автоматизации аудита, реализующих какой-либо стандарт.

Внутренний аудит проводится внутри сетевого пространства, ограниченного межсетевым экраном организации. Он также включает в себя сканирование портов и уязвимостей внутренних хостов организации. Кроме того, анализируются организация и выполнение установленной политики безопасности, контроль и управление доступом к ресурсам, парольная политика персонала организации и ее выполнение. Данный вид аудита дополняет стандартные методики проведения аудита более исчерпывающим рассмотрением сетевых уязвимостей.

1.6. Механизмы и службы защиты

Поскольку не существует одного устройства, обеспечивающего защиту от атак на целостность, конфиденциальность и доступность, используются комбинации компонентов защиты, что значительно уменьшает возможность успешной атаки. К основным компонентам безопасности относятся:

- межсетевые экраны;
- системы обнаружения вторжений;
- средства контроля целостности;
- средства проверки содержимого;
- сканеры;
- средства контроля конфигураций;
- средства контроля доступа и аутентификации;
- виртуальные частные сети (Virtual Private Network, VPN);
- встроенные средства безопасности (операционные системы, средства аудита, списки контроля доступа и т. д.).

Построение и применение основных компонентов безопасности будет рассмотрено далее.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Чем отличаются понятия «интранет» и «экстранет»?
2. Перечислите основные устройства межсетевого взаимодействия.
3. В чем сходства и отличия понятий угрозы и уязвимости?
4. Сформулируйте основные источники угроз.
5. Перечислите основные виды утечки информации.
6. В чем состоит сущность комплексного подхода к обеспечению информационной безопасности?
7. Сформулируйте назначение политики безопасности.
8. Перечислите основные шаги определения политики безопасности.
9. Перечислите основные шаблоны политики безопасности для известной организации.
10. Перечислите основные задачи сетевой политики безопасности.
11. Что такое сетевой периметр?
12. Сформулируйте определение демилитаризованной зоны.
13. Перечислите основные составляющие эшелонированной обороны.
14. Что такое управление рисками?
15. Сформулируйте составляющие общей оценки риска.
16. Каковы основные подходы к количественной оценке рисков?
17. Перечислите основные методы, применяемые для уменьшения риска.
18. Что такое остаточный риск?
19. Сформулируйте назначение и сущность аудита информационных систем.
20. Перечислите основные виды аудита и их особенности.
21. Перечислите основные механизмы и службы защиты.

Глава 2
ВРЕДОНОСНЫЕ ПРОГРАММЫ

— Если бы это было так, это бы еще ничего, а если бы ничего, оно бы так и было, но так как это не так, так оно и не этак! Такова логика вещей!

Л. Кэрролл. Алиса в стране чудес

Появление и широкое распространение компьютеров, особенно персональных, привело к появлению многочисленного класса вредоносных программ, т. е. программ, которые выполняют зловредные, а часто и деструктивные функции. Распространению таких программ способствовали: ознакомление с программированием широких масс, простота подключения к Интернету, появление переносимых носителей информации и возможность зарабатывания денег с помощью вредоносных программ.

Под *вредоносной программой* понимают компьютерную программу или переносимый код, которые предназначены для повреждения информации, хранимой в компьютерной сети, скрытого использования компьютерных ресурсов или для другого воздействия, которое нарушает нормальные операции компьютеров или компьютерной сети.

Различные источники оперируют множеством терминов для описания вредоносных программ. Приведем основные термины, используемые в англоязычной литературе: malware, spyware, grayware, virus, worm, rootkit, backdoor, logic bomb, PuP (Potentially unwanted Programs), spam, riskware, bot (botnet), zombie, BHO (Browser Helper Objects), trojan horse, RAT (Remote Administration Tool), RHT (Remote Hacker Tool), adware, phishig, vishing, pharming, rabbit, downloader, dialer, DDoS attacks, banking, exploiting, vulnerabilities, threats for mobile phones, IP-communication threats, social networking threats и т. д.

Согласно ГОСТ Р 51275 — 2006 вредоносной программой является «программа, используемая для осуществления несанкционированного доступа к информации и (или) воздействия на информацию или ресурсы автоматизированной информационной системы».

Специалисты конкретизируют это общее понятие следующим образом. Программой с потенциально опасными последствиями, или вредоносной программой, называют некоторую самостоятельную программу (набор инструкций), которая способна выполнять любое непустое подмножество следующих функций:

• обладать способностью к самодублированию, ассоциированию себя с другими программами и (или) переносу своих фрагментов в иные области оперативной или внешней памяти;

• разрушать (искажать произвольным образом) код программ в оперативной памяти;

• выполнять без инициирования со стороны пользователя (пользовательской программы в штатном режиме ее выполнения) деструктивные функции (копирование, уничтожение, блокирование и т. п.);

• сохранять фрагменты информации из оперативной памяти в некоторых областях внешней памяти прямого доступа (локальных или удаленных);

• искажать произвольным образом, блокировать и (или) подменять выводимый во внешнюю память или в канал связи массив информации, образовавшийся в результате работы прикладных программ, или уже находящиеся во внешней памяти массивы данных;

• скрывать признаки своего присутствия в программной среде компьютера.

Вредоносные программы могут быть внесены (внедрены) как преднамеренно, так и случайно в ПО, используемое в системе, в процессе его разработки, сопровождения, модификации и настройки. Кроме того, вредоносные программы могут быть внесены в процессе эксплуатации систем с внешних носителей информации или посредством сетевого взаимодействия (причем как в результате несанкционированного доступа, так и случайно пользователями).

Современные вредоносные программы основаны на использовании уязвимостей различного рода ПО (системного, общего, прикладного) и разнообразных сетевых технологий.

Общепринято выделять три основных вида вредоносных программ, отличающихся своим поведением и назначением:

1) компьютерные вирусы;

2) троянские кони (трояны);

3) сетевые черви.

Кроме этих видов часто используется понятие «программная закладка».

В соответствии с ГОСТ Р 51275—2006 программная закладка трактуется как «преднамеренно внесенный в программное обеспечение функциональный объект, который при определенных условиях инициирует реализацию недекларированных возможностей программного обеспечения. Примечание. Программная закладка может быть реализована в виде вредоносной программы или программного кода».

Необходимо отметить, что четкое разделение вредоносных программ на указанные виды практически невозможно, так как отдельные вредоносные программы часто выполняют и функции другого вида. Кроме того, вредоносные программы используют вспомогательные средства для скрытия своего присутствия и выполнения действий на инфицированной машине.

Все виды вредоносных программ предназначены для попытки нарушения одного из критериев безопасности информации: конфиденциальности, целостности и доступности.

Кратко рассмотрим основные виды вредоносных программ, их отличительные признаки и классификации.

2.1. Компьютерные вирусы

Первый компьютерный вирус появился в 1982 г. для Apple II. Сам термин «компьютерный вирус» впервые применил Фред Коэн (F. Cohen) в своей магистерской диссертации, руководителем которой был Леонард Адлеман (L. Adleman) — один из создателей алгоритма шифрования RSA, названного по первым буквам фамилий авторов (R. Rivest, A. Shamir, L. Adleman). Результаты исследований Коэна были опубликованы в журнале «Computers & Security» (1987. № 6), а в 1988 г. на конференции Crypto'88 Адлеман сделал доклад «An Abstract Theory of Computer Viruses», который был опубликован в 1990 г.

Определение компьютерного вируса, которое дал Ф. Коэн, таково: «Вирус — это программа, которая способна инфицировать другие программы путем их модификации, включая возможно измененную копию самой себя ((A virus is a program that is able to infect other programs by modifying them to include a possibly evolved copy of itself)». Более строгое определение дано Л. Адлеманом «Компьютерный вирус — программа, которая рекурсивно копирует возможно измененную версию себя».

Компьютер, на котором находится хотя бы один экземпляр компьютерного вируса (или другого вида вредоносного ПО), будем называть *инфицированным*.

Коэн привел псевдокод компьютерного вируса:

```
program virus :=
  { 1234567;
  subroutine infect-executable :=
    { loop: file = random-executable;
      if first-line-of-file = 1234567
        then goto loop;
      prepend virus to file;
    }
  subroutine do-damage :=
    { whatever damage is desired }
  subroutine trigger-pulled :=
    { return true on desired conditions }
main-program :=
  { infect-executable;
    if trigger-pulled then do-damage;
```

```
    goto next;
  }
  next:
}
```

В данном псевдокоде можно выделить основные функции компьютерного вируса: проверка на зараженность нового файла, проверка условия выполнения деструктивной нагрузки и инфицирование других файлов.

В нашей стране в 1987 г. появилась первая статья, посвященная компьютерным вирусам, опубликованная сотрудником вычислительного центра АН СССР А.А. Чижовым. Она базировалась на публикациях зарубежных авторов и предупреждала о серьезной опасности вирусов. В августе 1988 г. первый компьютерный вирус был обнаружен в лаборатории Института программных систем в Переяславле-Залесском после проведения международной олимпиады школьников.

Широкому распространению вирусов и их вариантов (штаммов) способствовало появление исходных текстов вирусов с подробными комментариями.

Начиная с 1989 г. в Киевском институте кибернетики им. А.М. Глушкова стал проводиться семинар «Системное программирование», который вел Н.Н. Безруков. В 1991 г. Н.Н. Безруков опубликовал книгу «Компьютерная вирусология», которая являлась энциклопедией компьютерных вирусов, их истории и основных подходов к обнаружению вирусов. По итогам семинара распространялись дискеты с первыми антивирусными программами, обнаруживающими и вылечивающими инфицированные последними вирусами файлы.

Определение компьютерного вируса было дано в ГОСТ Р 51188—98: «Компьютерный вирус — программа, способная создавать свои копии (необязательно совпадающие с оригиналом) и внедрять их в файлы, системные области компьютера, компьютерных сетей, а также осуществлять иные деструктивные действия. При этом копии сохраняют способность дальнейшего распространения. Компьютерный вирус относится к вредоносным программам».

В 1990 г. мировым лидером в области разработки и распространения файловых вирусов являлась Болгария. Но уже в 1991 г. был зафиксирован так называемый советский вирусный взрыв. Причинами этого явились следующие обстоятельства:

- наличие исходных кодов вирусов с комментариями;
- появление и широкое распространение персональных компьютеров;
- появление книг по программированию и обучение программированию;
- подключение к Интернету.

Обобщенный жизненный цикл компьютерного вируса включает в себя следующие этапы: внедрение (попадание на компьютер), инкубационный период (период времени, когда вирус не выполняет никаких своих функций для защиты от обнаружения), саморазмножение (инфицирование объектов компьютера), выполнение специальных функций (которые являлись целью создателя вируса) и проявление (это было характерно для ранней истории вирусов). Естественно, что компьютерные вирусы могут не содержать некоторых из перечисленных этапов. Например, в настоящее время вирусы, как правило, стараются никак не проявить себя, чтобы избежать обнаружения.

Рассмотрим простейшую классификацию кодов вирусов:

- степень опасности;
- среда обитания (программная, включая ОС, аппаратная);
- стратегия инфицирования;
- наличие маскировки (защита от просмотра, обнаружения).

По степени опасности (или деструктивным возможностям) компьютерные вирусы подразделяют на *безвредные*, или *неопасные* (не влияющие на работу компьютера, кроме уменьшения свободной памяти на диске в результате заражения файлов); *опасные*, которые могут привести к модификации инфицированных файлов или сбоям в работе компьютера, и *очень опасные*, в алгоритмах функционирования которых содержатся функции нанесения вреда (например, удаления файлов).

В зависимости от среды обитания компьютерные вирусы подразделяют на следующие виды:

- файловые;
- загрузочные (файлово-загрузочные);
- макровирусы;
- сетевые (вирусы, которые могут использовать некоторые способы для самостоятельного распространения, например, через адресную книгу инфицированного компьютера);
- заражающие объектные модули, библиотеки и исходные тесты программ.

Среди вирусов выделяют так называемые стелс-вирусы (stealth viruses) — вирусы-невидимки. Эти вирусы перехватывают обращения ОС, относящиеся к вызову инфицированных файлов или секторов жестких дисков, и в качестве результата подставляют неинфицированный код.

Стелс-вирусы могут использовать специальные методы защиты от обнаружения антивирусными программами и для противодействия отладчикам.

Вирусы, которые после запуска постоянно находятся в оперативной памяти, называют *резидентными*. Резидентные вирусы находятся в памяти и являются активными вплоть до выключения компьютера или перезагрузки ОС.

2.1.1. Файловые вирусы

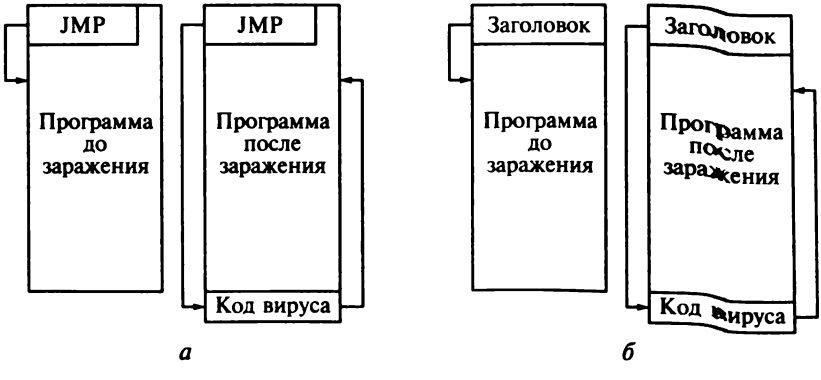
Файловые вирусы при своем размножении тем или иным способом используют файловую систему какой-либо ОС. По способу заражения файлов выделяют: перезаписывающие (overwriting), паразитические (parasitic), компаньон-вирусы (companion), link-вирусы, вирусы-черви (см. подразд. 2.3) и вирусы, заражающие объектные модули (OBJ), библиотеки компиляторов (LIB) и исходные тексты программ.

Метод заражения Overwriting является наиболее простым: перезаписывающий вирус записывает свой код вместо кода заражаемого файла, уничтожая его содержимое. Естественно, что при этом файл перестает работать и не восстанавливается. Такие вирусы очень быстро обнаруживают себя, так как ОС и приложения довольно скоро перестают работать.

К паразитическим относятся все файловые вирусы, которые при распространении своих копий обязательно изменяют содержимое файлов, оставляя сами файлы при этом полностью или частично работоспособными. Основными типами таких вирусов являются вирусы, записывающиеся в начало, середину или конец файлов. Пример записи вирусов в конец файлов приведен на рис. 2.1.

Отдельно следует отметить довольно незначительную группу паразитических вирусов, не имеющих «точки входа», — *ЕРО-вирусы* (Entry Point Obscuring viruses). Такие вирусы записывают команду перехода на свой код в какое-либо место в середину файла и получают управление не непосредственно при запуске зараженного файла, а при вызове процедуры, содержащей код передачи управления на тело вируса.

Компаньон-вирусы не изменяют заражаемые файлы. Алгоритм работы этих вирусов состоит в том, что для заражаемого



а *б*

Рис. 2.1. Схемы заражения:
а — COM-файла; *б* — EXE-файлов

файла создается файл-двойник, которому при запуске зараженного файла передается управление. Для систем Windows наиболее распространены компаньон-вирусы, использующие особенность DOS первыми выполнять файлы с расширением .bat, если в одном каталоге присутствуют три файла с одним и тем же именем, но различными его расширениями — .bat, .com и .exe. Такие вирусы создают для EXE-файлов файлы-спутники, имеющие то же самое имя, но с расширением .bat или .com. Если пользователь для запуска укажет только имя файла, то сначала выполнится вирус, который затем запустит требуемый файл. Другую группу компаньон-вирусов составляют вирусы, которые при заражении переименовывают файл и запоминают его, а свой код записывают под именем заражаемого файла. При запуске управление получает код вируса, который затем запускает оригинальную программу. Возможно существование и других типов компаньон-вирусов, использующих свои оригинальные идеи или особенности других ОС.

Файловые сетевые вирусы, кроме инфицирования программ на инфицированной машине, пытаются распространяться, используя имеющиеся прикладные программы (например, почтовые адреса).

Link-вирусы, как и компаньон-вирусы, не изменяют физического содержимого файлов, однако при запуске зараженного файла «заставляют» ОС выполнять свой код.

Вирусы, заражающие объектные модули, библиотеки компиляторов и исходные тексты программ, достаточно экзотичны и встречаются редко. В настоящее время их появление связано со все более широким применением языка Java.

Получив управление, файловый вирус совершает следующие общие действия:

- проверяет оперативную память на наличие своей копии и инфицирует память компьютера, если копия вируса не найдена (в случае, если вирус является резидентным), ищет незараженные файлы в текущем и (или) корневом каталоге путем сканирования дерева каталогов логических дисков, а затем заражает обнаруженные файлы;
- выполняет дополнительные (если они есть) функции: деструктивные действия, графические или звуковые эффекты и т.д. (эти дополнительные функции могут вызываться спустя некоторое время после активизации в зависимости от текущего времени, конфигурации системы, внутренних счетчиков вируса или других условий, в этом случае вирус при активизации обрабатывает состояние системных часов, устанавливает свои счетчики и т.д.);
- возвращает управление основной программе (если она есть).

К *сетевым* относятся вирусы, которые для своего распространения активно используют протоколы и возможности локальных и глобальных сетей. Основным принципом работы сетевого вируса является возможность самостоятельно передать свой код на удаленный сервер или рабочую станцию.

Обнаружить файловый нерезидентный вирус можно путем сравнения длин подозрительного файла и его дистрибутивной (или резервной) копии. Естественно, что данный метод не применим к обнаружению резидентного вируса.

2.1.2. Макровирусы

Макровирусы (macro viruses) являются программами на языках (макроязыках), встроенных в некоторые системы обработки данных (текстовые редакторы, электронные таблицы и т.д.). Для своего размножения такие вирусы используют возможности макроязыков и с их помощью переносят себя из одного зараженного файла (документа или таблицы) в другие. Наибольшее распространение получили макровирусы для программ Microsoft Office. Структура документа Microsoft Word приведена на рис. 2.2.

Для существования вирусов в конкретной системе (редакторе) необходимо наличие встроенного в систему макроязыка с возможностями:

- привязки программы на макроязыке к конкретному файлу;
- копирования макропрограмм из одного файла в другой;
- получения управления макропрограммой без вмешательства пользователя (автоматические или стандартные макросы).

Указанным условиям удовлетворяют прикладные программы Microsoft Word, Excel и Access. Они содержат в себе макроязыки: Word Basic, Visual Basic for Applications. При этом:

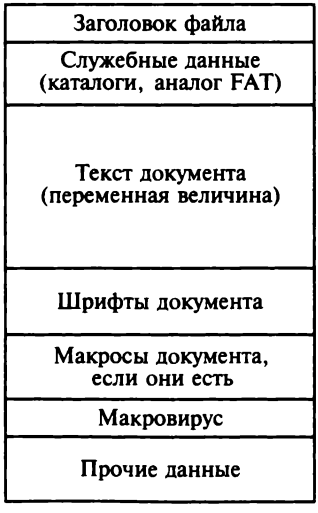


Рис. 2.2. Схема инфицированного документа

1) макропрограммы привязаны к конкретному файлу или находятся внутри файла;

2) макроязык позволяет копировать файлы или перемещать макропрограммы в служебные файлы системы и редактируемые файлы;

3) при работе с файлом при определенных условиях (открытие, закрытие и т.д.) вызываются макропрограммы (если таковые есть), которые определены специальным образом или имеют стандартные имена.

Возможности макроязыков таких систем позволяют автоматизировать обработку данных в больших организациях и наладить так называемый автоматизированный документооборот. Вместе с тем они позволяют вирусу переносить свой

код в другие файлы и таким образом заражать их, в частности инфицировать Excel, Power Point и Access макросам из Microsoft Word.

Характерными признаками проявлениями макровирусов в Microsoft Word являются:

- невозможность конвертирования зараженного документа Word в другой формат;
- появление файлов в формате Template (шаблон);
- невозможность записи документа в другой каталог или на другой диск по команде Save As и т. п.

Для проверки системы на предмет наличия макровирусов можно использовать пункт меню Tools/Macro. Если обнаружены «чужие» макросы, то они могут принадлежать вирусу. Однако этот метод не работает в случае стелс-вирусов, которые запрещают работу этого пункта меню, что в свою очередь является достаточным основанием считать систему инфицированной.

2.1.3. Загрузочные вирусы

Исторически загрузочные вирусы появились первыми, что было связано с необходимостью загружать ОС первых персональных компьютеров с дискет. Потом долгое время первенство по количеству занимали файловые вирусы, и только в последнее время загрузочные вирусы снова «стали на повестку дня». Этому в основном способствовало появление и широкое применение сменных носителей информации.

Рассмотрим типичную схему загрузки персонального компьютера (рис. 2.3).

Загрузочные вирусы могут записывать себя: в программу BIOS, в загрузочный сектор диска (boot-сектор), в сектор, содержащий системный загрузчик винчестера (master boot record), либо изменить указатель на активный boot-сектор. Они внедряются в память компьютера при загрузке с инфицированного диска, поэтому всегда

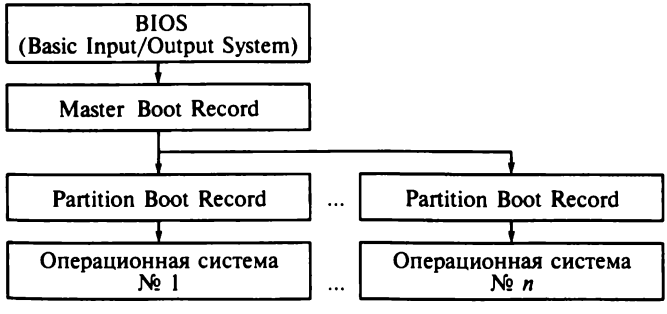


Рис. 2.3. Схема загрузки ОС

являются резидентными. Если код загрузочного вируса превышает размер сектора, то вирус помещает остаток кода в другие сектора жесткого диска, помечая их меткой bad. Операционная система не использует сектора с такой пометкой, что обеспечивает сохранность кода вируса.

При загрузке ОС системный загрузчик считывает содержимое первого сектора диска, с которого производится загрузка, помещает считанную информацию в память и передает на нее (т.е. на вирус) управление.

В дальнейшем загрузочный вирус ведет себя так же, как файловый: перехватывает обращения ОС к дискам и инфицирует их, в зависимости от установленных условий совершает специальные (иногда деструктивные) действия и может выполнять звуковые или видеоэффекты.

Для обнаружения загрузочного вируса можно воспользоваться мишенью — отформатированной на заведомо не зараженном компьютере дискетой. Если после использования на подозреваемом компьютере сравнение ее boot-сектора с оригинальным покажет отличия, то это свидетельствует о наличии на подозреваемом компьютере загрузочного вируса.

2.1.4. Методы защиты от обнаружения

В связи с развитием средств обнаружения и удаления вирусов (антивирусных средств) злоумышленники уделяют большое внимание разработке и применению различных методв защиты от обнаружения. Кратко перечислим некоторые из них:

- простейшая маскировка (вставка дополнительных команд — «мусора», не изменяющих функциональность вируса);
- усиление кода (замена команд на равнозначные, перестановка команд в программе, перестановка блоков команд, применение упаковщиков);
- шифрование (шифрование каждой копии вируса на своем ключе);
- вампиризм (формирование команд работы с файлами из команд, имеющихся на инфицированной машине);
- защита от антивирусов (удаление антивирусных средств, блокировка загрузки обновлений антивирусного провайдера).

Например, следующий вирусный код

```
mov eax, ebx
mov esi, edi
```

может быть заменен на

```
mov esi, edi
mov eax, ebx
```

Таблица 2.1. **Маскировка мутацией команд**

Код вируса до мутации	Код вируса после мутации
mov edi, 2580774443	mov ebx, 535699961
mov ebx, 67750807	mov edx, 1490897411
sub ebx, 1745609157	xor ebx, 2402657826
sub edi, 150468176	mov ecx, 3802877865
xor ebx, 875205167	xor edx, 3743593982
push edi	add ecx, 2386458904
xor edi, 3761393434	push ebx
push ebx	push edx
push edi	push ecx

(перестановка команд, не зависящих от данных). Другим примером может служить замена эквивалентных последовательностей:

```
push ebp
mov ebp, esp
```

заменяются на

```
push ebp
push esp
pop ebp
```

Пример мутации арифметических и логических команд приведен в табл. 2.1.

Применение этих и подобных методов защиты от обнаружения привело к появлению термина «полиформизм». Под *полиморфизмом*

Таблица 2.2. **Пример полиморфизма**

Одна из последующих копий	Другая из последующих копий

`mov edi, 5500000Fh`
`mov [esi], edi`
`pop edi`
`push edx`
`mov dh, 40`
`mov edx, 515EC0Bh`
`push ebx`
`mov ebx, edx`
`mov [esi+0004], ebx`
	`mov ebx, 5500000Fh`
	`mov [esi], ebx`
	`pop ebx`
	`push ecx`
	`mov ecx, 5FC000CBh`
	`add ecx, F191EBC0h`
	`mov [esi+0004], ecx`

компьютерного вируса понимается модификация кода в инфицированных файлах (объектах) без изменения его функциональности. Полиморфные вирусы не имеют сигнатур, поэтому в большинстве случаев два образца одного и того же «полиморфик»-вируса не будут иметь ни одного совпадения. В качестве примера полиморфизма покажем, как строки кода вируса W32/Evol

```
mov dword ptr [esi], 5500000Fh
mov dword ptr [esi+0004], 5151EC0Bh
```

изменяются при очередных инфицированиях (табл. 2.2).

Полиформизм также реализуется шифрованием основного тела вируса и модификациями программы-расшифровщика.

В последние годы наблюдается появление первых образцов *метаморфных вирусов*, т. е. вирусов, которые при каждом инфицировании не только модифицируют свой код, но и изменяют свою функциональность.

2.2. Троянские кони

Основным источником новых программ в 80-е гг. XX в. были так называемые доски объявлений (Bulletin Board System, BBS). Первые появления вредоносных программ на BBS зафиксированы в 1985 г., на одной из таких досок был размещен список имен 12 зловредных программ, который назвали «грязная дюжина» (Dirty dozen). Это название прижилось, поэтому подобный список Dirty dozen в 1987 г. содержал уже 166 имен. Приведем пример одного из описаний из списка, содержащего употребление термина «троян»:

«TROJAN (T) This program PURPOSEFULLY damage a user's system upon their invocation. They almost always shoot to disable hard disk, although they can, in rare cases, destroy other equipment too. There are many ways that a TROJAN can disable your hard disk.»

Данное название, заимствованное из истории древней Греции, прижилось и стало применяться ко всем программам, которые содержали вредоносную функцию.

Троянский конь (или просто *троян*) — это программа, которая кроме объявленной функции, содержит скрытую, вредоносную функцию. Очень часто троянов называют программными закладками. Чтобы не путать эти понятия, будем называть программной закладкой троянского коня, введенного разработчиком программы. Например, на сайте www.eggs.com содержится громадная коллекция различных программных закладок для всех ОС и множества приложений, которые внедрялись в них разработчиками. Эта коллекция постоянно пополняется пользователями различных стран.

Для начала функционирования на новом компьютере троян должен быть запущен пользователем (запущена троянизированная про-

грамма). Отметим, что инфицированная компьютерным вирусом программа, по сути, является троянским конем. После запуска троянская программа удаляет себя из файла-носителя, принимает меры к обеспечению своей выживаемости при перезагрузке и к постоянному функционированию на компьютере, применяя различные методы маскировки от обнаружения. Таким образом, троянский конь существует на инфицированном компьютере в одном экземпляре и является постоянно выполняющимся процессом.

Приведем классификацию троянских коней «лаборатории Касперского», выделив их в соответствии с основными угрозами (рис. 2.4).

По критерию конфиденциальности можно выделить:

- Backdoors (потайные ходы). Эти программы позволяют злоумышленнику обходить нормальный контроль безопасности входа и дают ему доступ к компьютеру-жертве;
- Trojan-PSW. Копируют (или перехватывают) вводимые пользователями пароли и передают их злоумышленнику;
- Trojan-Spy — шпионские программы. Название данного вида троянских программ указывает на преследуемую ими цель — шпионаж за действиями пользователя инфицированного компьютера. Этот шпионаж может включать в себя сбор вводимой с клавиатуры информации, снятие снимков экрана компьютера, копирование

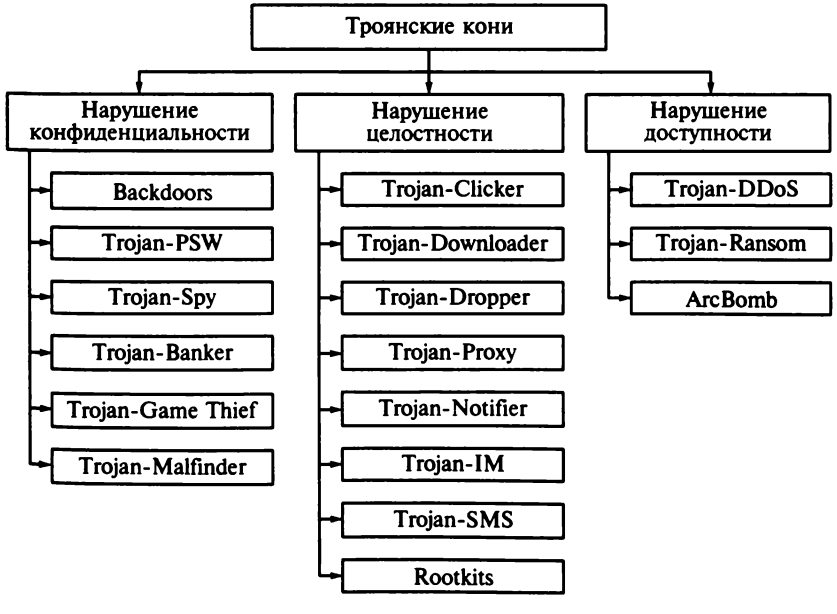


Рис. 2.4. Классификационная схема троянских коней

списка активных приложений и т. п. Обычно такие программы используются для получения паролей пользователя и других данных, необходимых для осуществления интернет-платежей, и данных для выполнения банковских операций. Собранные таким образом данные или передаются автору троянской программы, или хранятся на инфицированном компьютере, ожидая проникновения в инфицированный компьютер для получения этих данных (чаще всего это реализуется установлением потайного хода на инфицированном компьютере). В связи с особой опасностью для пользователей из данного класса троянских коней отдельно выделен класс Trojan-Banker.

- Trojan-Banker. Собирают информацию пользователя, относящуюся к проведению банковских операций. Эта информация также отправляется злоумышленнику;
- Trojan-Game Thief. В связи с широким распространением компьютерных игр многие пользователи стремятся купить дополнительные услуги или бонусы для повышения своего уровня. Данные программы собирают такие данные и передают злоумышленнику, который использует их для продажи;
- Trojan-Mailfinder. Собирают почтовые адреса из почтовой службы пользователя и пересылают их злоумышленнику, который может использовать их для рассылки спама.

По критерию целостности различают:

- Trojan-Clicker. Перенаправляют браузер пользователя на заданные злоумышленником сайты или другие ресурсы Интернета;
- Trojan-Downloader. Программы загрузки новых версий вредоносных программ и их инсталляции на инфицированном компьютере (постоянно просматривают заданные зловредные сайты и загружают на инфицированный компьютер новые версии вредоносного ПО).
- Trojan-Dropper. Предназначен для скрытной инсталляции вредоносных программ на компьютер пользователя;
- Trojan-Proxy. Функционирует как прокси-сервер и предоставляет анонимный доступ в Интернет из компьютера пользователя;
- Trojan-Notifier. Сообщает злоумышленнику данные новой инфицированной машины;
- Trojan-IM. Определяет и передает злоумышленнику данные учетной записи пользователя для интернет-пейджера (ICQ, MSN Messenger, AOL Instant Messenger, Yahoo Pager, Skype и т. п.);
- Trojan-SMS. Служит для неавторизованной рассылки SMS-сообщений из инфицированных мобильных устройств на платные услуги;
- Rootkits. Набор средств, позволяющих злоумышленнику скрыть наличие и выполнение вредоносных программ на инфицированной машине.

По критерию доступности троянских коней подразделяют на следующие виды:

• Trojan-DDoS. Набор средств для проведения удаленных атак «отказ в обслуживании» с инфицированной машины (их часто называют ботами), которые ожидают команды на начало атаки заданной цели или посылку множества писем на заданную цель;

• Trojan-Ransom. Предназначен для модификации данных на инфицированной машине (обычно с использованием шифрования) и требованием выкупа для их расшифрования;

• ArcBomb. Эти программы (архивированные файлы) предназначены для вывода из строя машины, пытающейся разархивировать данный файл.

Статистика последних лет показывает, что троянские кони являются преобладающим видом вредоносного ПО. В силу особенностей их применения отдельные виды троянских коней, а именно потайные ходы, руткиты и средства сетевых атак, далее будут рассмотрены подробнее.

2.3. Сетевые черви

Сетевым червем будем называть программу, которая самостоятельно перемещается по узлам сети (осуществляет самостоятельный выбор адреса цели и перемещение в него).

Практическая история сетевых червей началась с понедельника 2 ноября 1988 г., когда во многих компьютерах университетов и лабораторий США была обнаружена полная загруженность процессоров, занятых выполнением каких-то задач. Загрузка процессоров превышала все нормы и не позволяла пользователям и даже администраторам получить доступ к компьютерам. Перезагрузка компьютеров не давала результатов, поскольку через достаточно небольшой промежуток времени компьютеры снова оказывались загруженными различными задачами. Этот день был позже объявлен «черным понедельником», «атакой нового вируса» (термин «червь» еще не использовался). В ночь на вторник специалисты уже нашли одну уязвимость, использованную червем, но сообщения о противоядии не дошло до получателей вследствие загрузки сетей и компьютеров.

4 ноября 1988 г. было объявлено о полном дизассемблировании червя, кроме того, сам автор, Роберт Таппан Моррис, явился с повинной.

Рассмотрим основные модули червя и их функционирование. Сначала червь просматривает файлы на инфицированной машине, чтобы определить адреса соседних машин и машин, с которыми инфицированная машина поддерживает доверительные отношения. Далее червь ищет удаленные учетные записи с тем же именем, под которым функционирует червь на местной машине. Если таких записей не найдено, то пробуется выполнение утилиты finger. Если же и ее выполнение не дает результата, выполняется программа send-

mail (в ней используется метод переполнения буфера). Общая схема функционирования червя в псевдокоде выглядит следующим образом (весь код червя составлял 300 строк на языке С):

```
static mainloop()
{
    long key, time1, time0;
    time(&key);
    srandom(key);
    time0 = key;
    if (hg() == 0 && hl() == 0)
    ha();
    checkother();
    report_breakin();
    cracksome();
    other_sleep(30);
    while (1) {
  /* Crack some passwords */
       cracksome();
  /* Change my process id */
  if (fork() > 0)
      exit(0);
  if (hg() == 0 && hi() == 0 && ha() == 0)
      hl();
  other_sleep(120);
  time(&time1);
  if (time1 - time0 >= 60*60*12)
     h_clean();
  if (pleasequit && nextw > 0)
     exit(0);
  }
}
```

Червь применял множество методов для защиты от обнаружения, в том числе:

- при старте — обнуление списка аргументов;
- установка имени процесса — sh;
- периодическая смена идентификатора процесса, используя команду fork();
- удаление принятых файлов с жесткого диска (весь червь находился в оперативной памяти);
- наложение гаммы (XOR с константой) на все текстовые строки в программах червя;
- запрет core dump, чтобы избежать выдачи содержимого памяти на внешнее устройство.

Еще одной особенностью червя являлось то, что он достаточно успешно подбирал пароли на инфицированных машинах. Для этого

Моррис применял несколько методов. Первый метод состоял в использовании различных комбинаций букв, полученных из учетного имени пользователя. Например, из учетной записи Bill Clinton осуществлялись попытки использовать Bill, Clinton, Clinton Bill, clintonbill, billclinton, llibnotnilc и т. п. Второй метод заключался в проверке совпадения пароля с заданным списком типичных паролей из 432 слов (предварительно подготовленных Моррисом). Последним методом являлся перебор слов стандартного словаря (/usr/dict/words), который предназначался в Unix-системах для проверки правописания. Поскольку создателю червя необходимо было при каждой попытке шифровать слово, Моррис использовал свою утилиту шифрования crypt(), которая, хотя и была в три раза длиннее стандартной (6 Кбайт), но работала в девять раз быстрее.

Поскольку каждая инфицированная машина могла получить червя с другой инфицированной машины, Моррис реализовывал специальную процедуру для того, чтобы на машине оставался только один экземпляр червя, но с вероятностью 1/7 позволял пришедшему червю также использовать уже инфицированную машину. Это и привело к тому, что на каждой инфицированной машине выполнялось множество копий червя, что приводило к максимальной загрузке процессоров инфицированных машин.

На суде в 1989 г. были оглашены потери: инфицировано около 6 200 машин (7,3 % всех машин — через Интернет), общие потери составили около 98 млн долл. (работа программистов и администраторов, затраты машинного времени и потеря доступа). Приговор был достаточно мягок: три года испытательного срока, 400 ч общественных работ и 10 000 долл. штрафа.

В настоящее время Роберт Т.Моррис работает и преподает в MIT Computer Science and Artificial Intelligence Laboratory.

Последующие черви появились только через 10 лет. Причем их функциональность только копировала идеи Морриса. Основными особенностями сетевых червей являются: платформа, методы выбора цели, методы проникновения (инфицирования), количество используемых уязвимостей и методы маскировки (рис. 2.5).

Кратко рассмотрим основные механизмы сетевого червя:

1. Выбор цели. Заранее подготовленные списки адресов, почтовые адреса из адресной книги, сетевые соседи, доверенные системы, случайный выбор адреса, запросы DNS.

Рис. 2.5. Основные механизмы сетевых червей

2. Поиск уязвимости. Сканирование цели, идентификация операционной системы и прикладного ПО, посылка пакетов, использующих уязвимость.

3. Проникновение. Код, который использует известную злоумышленнику уязвимость (переполнение буфера, совместное использование файлов, посылка электронной почты с приложением, использование ошибок конфигурирования и т. д.).

4. Распространение. После получения доступа к машине необходимо получить основное тело червя, для чего используются протоколы FTP, TFTP, HTTP, SMB и др.

5. «Полезная нагрузка» (набор кодов, разработанных для выполнения определенных действий в интересах злоумышленника на целевой машине). Некоторые виды «полезной нагрузки» червя: установка потайного хода, установка бота, программы выполнения сложных математических операций, установка троянского коня и т. п.

Лаборатория Касперского представляет следующую классификацию червей: Email-Worm (черви, использующие почтовые системы для распространения), IM-Worm (использующие интернет-пейджеры), IRC-Worm (использующие протокол IRC), Net-Worm (собственно сетевые черви, использующие для распространения стек протоколов TCP/IP), P2P-Worm (черви, использующие одноранговые сети), Worm (черви, не попавшие в предыдущие категории).

Можно выделить некоторые тенденции в развитии современных сетевых червей:

- многоплатформенность (способность функционировать в различных операционных средах);
- использование множества уязвимостей;
- принятие специальных мер для увеличения начальной скорости распространения (заранее подготовленные списки адресов и т. п.);
- использование уязвимостей zero-day (т. е. уязвимостей, которые еще не были известны);
- наличие модулей обхода или блокировки средств защиты;
- изощренные методы маскировки присутствия и выполнения.

2.4. Потайные ходы

Потайной ход — это программа, которая позволяет злоумышленнику обходить нормальный контроль безопасности входа и дает ему доступ к компьютеру-жертве. Потайной ход может давать злоумышленнику различные виды доступа:

- локальная (местная) эскалация привилегий;
- удаленное выполнение команд на машине-жертве;
- удаленный доступ к командной строке машины-жертвы;
- удаленное управление компьютером-жертвой с использованием GUI;
- выдача специальных команд средству удаленных атак.

Установка потайного хода на машину-жертву может быть осуществлена в результате получения доступа к этой машине, путем автоматической инсталляции сетевым червем или троянским конем или обусловлена заинтересованностью пользователя в запуске программы (методы социальной инженерии), содержащей потайной ход. Потайной ход должен обеспечить свое постоянное функционирование и выживаемость после перезагрузки.

Для обеспечения выживаемости могут быть использованы следующие возможности ОС Windows:

- автоматический старт программ;
- вставка в файлы и папки Startup;
- установка соответствующих ключей реестра;
- использование мастера планирования заданий.

Как правило, потайной ход открывает заранее заданный порт, ожидая команд злоумышленника. Поэтому одним из методов обнаружения потайных ходов является анализ открытых портов.

2.5. Руткиты

Руткит (rootkit) — это набор средств, предназначенных для сокрытия присутствия и выполнения объектов в системе. Обычно скрываются файлы, папки, выполняющиеся процессы, открытые порты, значения ключей реестра.

Руткиты появились в начале 90-х годов XX в. для ОС SunOS и первоначально использовались сетевыми администраторами. Позже они стали распространяться среди хакерской элиты. В настоящее время существует множество различных руткитов для различных ОС. Кроме того, значительная часть руткитов может быть загружена из Интернета.

Различают два основных вида: ругкиты уровня пользователя (user-mode rootkits) и руткиты уровня ядра (kernel-mode rootkits).

2.5.1. Руткиты уровня пользователя

Руткиты уровня пользователя первоначально появились для Unix-систем и содержали модули:

- перезаписи двоичных файлов для доступа к потайному ходу;
- перезаписи двоичных файлов для маскировки присутствия;
- средств маскировки, не перезаписывающих существующие программы (например, удаление следов вторжения в записях аудита);
- дополнительных средств (снифферы, наборы «имя/пароль» и т. п.);
- скриптов инсталляции rootkit.

Возможность таких действий определяется доступностью исходных текстов многих ОС. Например, руткит LRK (Linux RootKit) имеет следующие свойства:

- мощные средства, перезаписывающие важные Linux-программы (наличие исходных текстов);
- изменяемость состава (постоянное добавление функциональности к основным средствам);
- непрерывное развитие во времени;
- доступность и простота использования.

Появление средств обнаружения руткитов привело к тому, что злоумышленники стали концентрироваться на более глубоком проникновении в ОС.

2.5.2. Руткиты уровня ядра

Большинство процессоров современных компьютеров включает в себя средства, позволяющие ПО системы функционировать на различных уровнях привилегий. На процессорах x86 эти уровни называются *кольцами* (от 0 до 3). Для ОС Linux и Windows используются только кольца 0 и 3. Само ядро ОС выполняется в кольце 0, а все процессы пользователя — в кольце 3. Упрощенная схема ОС Microsoft Windows приведена на рис. 2.6.

Основными функциями ядра являются функции управления:

- управление процессами и потоками — ядро указывает на выполняющийся процесс, создает процессы и потоки внутри процесса. Выделяет память для выполнения;
- взаимодействием процессов — если процессу необходимо передать данные другому процессу или ядру, он может использовать различные средства взаимодействия процессов;

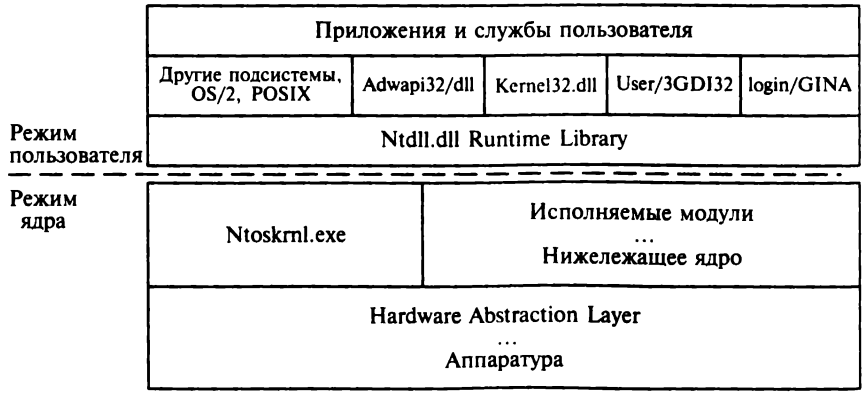


Рис. 2.6. Упрощенная схема ОС Windows

• памятью — ядро назначает память для запущенных программ и освобождает память. Использует функции управления виртуальной памятью;
• файловой системой — ядро управляет всем доступом к жестким дискам;
• аппаратурой — ядро контролирует интерфейсы между различными элементами аппаратуры (клавиатура, мышь, видео, аудио), чтобы различные программы могли их использовать для ввода/вывода;
• прерываниями — прерывания вызывают вызов кода ядра или посылку информации определенному процессу.

Руткиты уровня ядра используют следующие основные методы (для ОС Linux и Windows):
• применение зловредных драйверов устройств;
• внедрение в работающее ядро в памяти;
• перезапись образа ядра на жесткий диск;
• применения ядра виртуальной системы для обмана пользователей;
• запуск кода режима пользователя на уровне ядра.

Для ОС Linux можно выделить пять основных способов применения руткитов уровня ядра:
1) злонамеренные загружаемые модули;
2) изменение содержимого директории /dev/kmem;
3) установка патча для образа ядра на жестком диске;
4) создание ложного обзора системы;
5) изменение самого ядра.

Для ОС Windows основными методами использования руткитов уровня ядра являются:
• использование существующего интерфейса для вставки зловредного кода между существующими файлами;
• перезапись существующих исполняемых файлов;
• помещение зловредного кода в память исполняемых процессов с помощью DLL injection, API hooking.

Обнаружить установленный руткит достаточно трудно. Для этой цели рекомендуется использовать загрузочный CD-ROM со средствами анализа вредоносных программ. Например, компания F-Secure (http://www.f-secure.com/en/web/labs_global/removal/rescue-cd) предоставляет такой диск (с ОС Knoppix (производной от Linux) для проверки систем с ОС Windows.

2.6. Вредоносные программы для мобильных устройств

Мобильные устройства подразделяют на мобильные телефоны (работают под управлением прошивки), смартфоны (используют ОС, но не имеют тачскрина) и коммуникаторы (используют ОС, но

содержат сенсорный экран). Основными ОС для коммуникаторов и смартфонов являются Symbian, Windows Mobile и Google Android.

Годом появления новой угрозы информационной безопасности стал 2004 год. В июне был обнаружен первый вредоносный код (Worm.SymbOS.Cabir.a), поражающий мобильные телефоны, работающие под управлением ОС Symbian. Публикация в Интернете исходных кодов Cabir привела к возникновению ряда модификаций данного зловредного кода (которые были выявлены в 20 различных странах). Этот червь был практически безвреден — он лишь рассылал сам себя по Bluetooth, расходуя тем самым заряд аккумулятора устройства. Уже через месяц появился *вирус Duts* — первый вирус для платформы Windows Mobile. Этот вирус имел способность заражать собой исполняемые файлы, однако перед заражением спрашивал разрешения у пользователя КПК или коммуникатора.

Следующим для Windows Mobile явился *вирус Brador* — первый потайной ход для мобильной платформы. Brador ожидал подключения зараженного устройства к Интернету, и как только оно было установлено, он отправлял IP-адрес устройства «хозяину» по e-mail и открывал для него порт. Злоумышленник (хозяин), подключившись через этот порт к инфицированному устройству, мог получить доступ к его файлам.

Для этой же платформы появился следующий вирус — *Mosquit*, явившийся первым мобильным трояном. Вирус Mosquit возник в результате внедрения вредоносного кода в первоначально безобидную игру Mosquitos. Инфицированная игра при запуске отправляла SMS-сообщения на короткие номера злоумышленника, тем самым принося ему доход. Данные представители и их аналоги формируют следующие основные типы вредоносных программ для мобильных устройств:

- черви, распространяющиеся через специфические протоколы и сервисы;
- троянцы-вандалы, использующие ошибки ОС для установки в систему;
- троянцы, ориентированные на нанесение финансового ущерба пользователю.

Наиболее опасными являются черви, способные вызвать быстрое заражение большого количества систем, нарушить работоспособность мобильной сети или превратить ее в подконтрольную злоумышленнику распределенную сеть ботов (сеть зомби).

Мобильные вирусы — это компьютерные (программные) вирусы, разработанные злоумышленниками специально для распространения через мобильные устройства. Чаще всего мобильные вирусы распространяются с помощью SMS- и MMS-сообщений, а также по каналу Bluetooth. Основными причинами распространения мобильных вирусов являются:

- уязвимости ПО;

• низкий уровень «мобильной» грамотности;

• отношение владельцев мобильных телефонов к мобильным вирусам как к проблеме будущего;

• несоблюдение элементарных правил безопасности.

В настоящее время существует несколько путей проникновения вирусов в мобильные устройства: с другого телефона через Bluetooth-соединение, посредством MMS-сообщения, из персонального компьютера (соединение через Bluetooth, Wi-Fi, USB и т. п.), через Web- или Wap-сайты. Ресурсы мобильных устройств на данный момент несопоставимы с ресурсами персональных компьютеров, что определяет особенности мобильных вирусов и их отличия от обычных компьютерных вирусов:

• антивирусные средства для КПК содержат вирусную базу, включающую в себя только вирусы для мобильных устройств. Это вызвано ограниченностью ресурсов мобильных устройств;

• использование антивирусными средствами ресурсов КПК снижает временной промежуток между зарядками аккумуляторов;

• в настоящее время разнообразие и количество вирусов для мобильных устройств еще не очень велико (по сравнению с вирусами для персональных компьютеров).

На выбор злоумышленниками мобильной платформы влияют два основных фактора: распространенность ОС и ее открытость. Лидером по обоим критериям в настоящее время стала ОС Android. Именно потому интерес вирусописателей и направлен в большей степени на данную платформу. В 2011 г. были зафиксированы вредоносные программы в официальном магазине приложений Android. Сегодня количество вредоносного для Android обеспечения составляет около 59 % общего количества вредоносного ПО для мобильных устройств.

Количество мобильных вирусов резко увеличилось, когда функциональность смартфонов сравнялась с ноутбуками и домашними персональными компьютерами. Выход в Интернет, работа с почтой, интеграция в социальные сети и мобильный банкинг способствовали привлечению внимания злоумышленников.

Рост числа и степени опасности вредоносных программ для мобильных платформ привел к появлению мобильных антивирусов. Базовый набор функций мобильного антивируса включает в себя проверку системы в реальном времени, сканирование по требованию и механизмы для безопасной работы в Интернете. Практически все антивирусные компании предлагают средства защиты мобильных устройств. Например, антивирус «Лаборатории Касперского» Mobile работает в большинстве мобильных ОС и предоставляет пользователям всестороннюю защиту с помощью периодического сканирования и постоянного мониторинга (антивирусный резидентный монитор), а компания «Доктор Веб» предлагает и свободно распространяемую версию антивируса (http://download.drweb.com/android).

В настоящее время можно отметить следующие изменения в мире мобильных вирусов:

- расширение списка платформ, для которых зафиксированы вредоносные программы (добавились iOS — ОС для iPhone/iPod Touch/iPad и Android);
- изменение соотношения различных ОС для мобильных устройств;
- усложнение вредоносных программ и атак. Злоумышленники стали использовать различные сочетания уже известных технологий, например, за счет широкого распространения мобильного Интернета вредоносные программы получили возможность взаимодействовать с удаленными серверами злоумышленников и получать от них обновления и команды. Такие возможности уже используются преступниками для создания сетей мобильных ботов;
- нацеленность вредоносных программ на получение финансовой прибыли злоумышленниками.

2.7. Прочие вредоносные программы

К числу *прочих вредоносных программ* относят разнообразные программы, которые не входят в основные классы: вирусы, черви и трояны. Программы, предназначенные для проведения удаленных сетевых атак и атак «отказ в обслуживании», рассмотрены в гл. 3. Другим классом таких программ являются хакерские утилиты, позволяющие взламывать другие компьютеры (exploits, hacker tools). Эти программы дают возможность проникновения в удаленные компьютеры в целях дальнейшего управления ими (устанавливая на инфицированном компьютере потайной ход) или установки других вредоносных программ. Эксплойты используют известные или найденные злоумышленником уязвимости в ОС или приложениях.

Генераторы (constructors) — это конструкторы вирусов, троянских программ и червей, предназначенные для изготовления новых образцов вредоносных программ. Такие генераторы позволяют генерировать исходные тексты вирусов, объектные модули и (или) непосредственно зараженные файлы. Особой разновидностью являются *полиморфные генераторы*, которые обеспечивают зашифрование и расшифрование кода вируса и включаются в состав других видов вредоносных программ.

Программы дозвона на платные ресурсы (dialers) не наносят вреда компьютеру, но могут привести к финансовому ущербу для владельца номера, с которого реализуется дозвон на платный ресурс. Различают два типа программ дозвона. Один разработан для скрытной установки в систему и автоматического дозвона на платные ресурсы, т.е. используется в мошеннических целях. Второй информи-

рует пользователя о предстоящих действиях, сообщает тарифы, имеет корректную процедуру деинсталляции и т. п.

Программы-загрузчики (downloaders) предназначены для загрузки из сети и установки на компьютер ПО. Зловредные загрузчики осуществляют скачивание загруженных программ, как правило, скрытно от пользователя, выполняя эти операции одновременно с аналогичными операциями пользователя. После этого производится инсталляция загруженных программ.

Под *спамом* понимают анонимную массовую незапрошенную рассылку почтовых сообщений. Анонимность означает отсутствие действительного адреса отправителя или фальсификация его. Массовость является главным признаком спама и его отличительной чертой, благодаря которой спам существует и продолжает развиваться. Например, массовая рассылка рекламы может поднять престиж и финансовый успех компании. В настоящее время спам составляет около 80 % всех электронных сообщений в мире. Незапрошеная рассылка означает, что получатель не запрашивал источника о посылке сообщения.

Отдельную категорию составляют *мошеннические письма* (например, так называемые нигерийские письма с просьбой помочь нуждающимся), а также письма, направленные на кражу паролей, персональных данных и номеров кредитных карт (так называемый фишинг, phishing).

Шпионскими программами (spyware) называют семейство программ, предназначенных для хищения персональных и конфиденциальных данных пользователя. Типичными представителями этого класса являются *клавиатурные шпионы* (keyloggers), перехватывающие набираемые пользователем символы и пересылающие собранные данные автору (хозяину) клавиатурного шпиона.

Рекламные программы (adware) демонстрируют рекламные сообщения, могут подменять результаты поиска и др. Очень часто рекламные программы выполняют и функции шпионских программ, поэтому грань между «безобидными» рекламными программами и полноценными вредоносными программами практически стерта. Все больше и больше обнаруживаемых программ этого класса содержат в себе черты троянцев. Эти программы всячески стараются затруднить свое обнаружение и деинсталляцию из системы, они ищут и удаляют программы-конкуренты, сами занимая их место. Они могут содержать модули для сбора и отправки третьим лицам информации о посещаемых сайтах и вводимых владельцем компьютера данных.

Browser Hijackers — это программы, которые замещают просмотр базовой или искомой страницы, результаты поиска или сообщения об ошибке другими данными, которые пользователь не запрашивал. Этот тип вредоносных программ относят к adware.

Программы-шутки (Bad-Joke, Hoax) — злые шутки, вводящие пользователей в заблуждение относительно наличия вредоносных

программ. К ним относятся программы, которые не причиняют компьютеру какого-либо прямого вреда, однако выводят сообщения о том, что такой вред уже причинен либо будет причинен при каких-либо условиях, либо предупреждают пользователя о несуществующей опасности. К «злым шуткам» относятся, например, программы, которые «пугают» пользователя сообщениями о форматировании диска (хотя никакого форматирования на самом деле не происходит), детектируют вирусы в незараженных файлах, выводят странные вирусоподобные сообщения и т.д.

2.8. Наименование вирусов

Исторически сложилось так, что практически все вирусы имеют свои имена. Это началось с первого вируса El Cloner, который получил свое имя от текста, который он выдавал на экран.

Общепринятым является подход, когда имя вируса (и других образцов вредоносных программ) строится по следующему правилу:

Префикс.Имя.Суффикс

Поле *Префикс* обозначает платформу распространения вируса или тип вируса.

Имя, как правило, извлекается из тела вируса.

Суффикс предназначен для различения вариантов вирусов одного семейства.

В настоящее время компания «Лаборатория Касперского» установила новую систему наименования вирусов, структура которого выглядит следующим образом:

Поведение.Платформа.Имя.[Вариант]

Поле Поведение устанавливает вид детектируемого объекта по его поведению. Для вирусов и червей поведение определяется по способу распространения; троянских программ и вредоносных утилит — по совершаемым ими действиям; потенциально опасных программ (Potentially Unwanted Programs, PUP) — по функциональному назначению детектируемого объекта. Для отдельных видов поведения может указываться тип поведения, который записывается через дефис, например, P2P-Worm, Net-Worm, Trojan-Dropper.

Поле *Платформа* определяет среду выполнения программного кода. Платформа может быть как программной, так и аппаратной. Если детектируемый объект использует множество платформ, то реализуется платформа с наименованием Multi. Примеры платформ: Win32, BAT, IRC и т.п.

Поле *Имя* используется для названия детектируемого объекта или его семейства. Если объект не подпадает под известные семейства,

то ему присваивается обобщенное имя, например Trojan.Win32.Dialer. Имя обычно дается, исходя из текстовых строк, присутствующих в детектируемом объекте. Например, червь, содержащий текстовую строку «Bucheon», получил соответствующее имя. Имя червя Gavir образовано из текстовой строки «c:\gamevir.txt». Червь Skybag получил это наименование, так как являлся смесью червей NetSky и Bagle.

Поле *Вариант* отсутствует, если это единственный экземпляр (первый образец объекта). Обычно поле варианта содержит буквы, начиная с *a* до *z*, далее с *aa* до *zz* и т.д. Например, Bagle.a, Bagle.b и т.д.

Поскольку имя дает антивирусная компания, которая получила экземпляр инфицированного файла, то часто один и тот же вирус имеет разные наименования. Это часто приводит к неоднозначности наименований, которые даны одному и тому же вирусу. Наименования различных вредоносных объектов приводятся на сайте *www.wildlist.org* (The WildList Organization International), которые ведут WildList-списки:

"The WildList is currently being used as the basis for in-the-wild virus testing and certification of anti-virus products by the ICSA, Virus Bulletin and Secure Computing".

2.9. Элементы защиты от вредоносного программного обеспечения

В 1984 г. в своей работе «Computer Viruses — Theory and Experiments» Ф.Коэн показал, что задача обнаружения компьютерных вирусов является неразрешимой. При этом он руководствовался следующими рассуждениями.

Компьютерным вирусом, как определил Коэн, является программа, способная заражать другие программы, изменяя их таким образом, чтобы они включали возможно измененную копию вируса. Следовательно, чтобы определить, является ли программа *P* вирусом, необходимо выяснить, заражает ли она другие программы. Предположим, существует процедура *D*, позволяющая по любой программе *P* сказать вирус это или нет. Но тогда можно составить программу *V*, которая будет включать процедуру *D* и заражать другие программы только в том случае, когда *D* не определит *V* как вирус, и не будет заражать в противном случае.

Соответственно, если процедура *D*, будучи примененной к программе *V*, скажет, что это вирус, программа *V* на самом деле не будет заражать файлы и не будет вирусом согласно определению. И наоборот, если согласно процедуре *D* программа *V* не является вирусом,

на самом деле она будет заражать другие программы, а значит, будет вирусом.

Таким образом, гипотетическая процедура *D* оказывается противоречивой и, следовательно, не существует.

Необнаружимость компьютерных вирусов означает то, что практически невозможно защититься от нового компьютерного вируса. Это усугубляется тем обстоятельством, что автор вируса может проверить его необнаруживаемость текущими версиями существующих антивирусных средств.

Рассмотрим некоторые признаки, которые могут говорить о том, что компьютер поражен (инфицирован):

- программы выполняются медленнее, чем обычно;
- компьютер перестает реагировать на клавиатуру или периодически блокирует ее работу;
- компьютер самопроизвольно перезагружается;
- появляются необычные сообщения об ошибках;
- меню и диалоговые окна отображаются в искаженном виде;
- антивирусную программу невозможно запустить или невозможно получить обновления для нее;
- на рабочем столе появились новые ярлыки;
- некоторые приложения (программы) исчезли.

Такие признаки при функционирующем антивирусном программном обеспечении свидетельствуют о том, что компьютер инфицирован еще неизвестным вирусом. Интервал возможного заражения компьютера пользователя новым видом вредоносной программы может составлять до нескольких дней и даже недель (рис. 2.7).

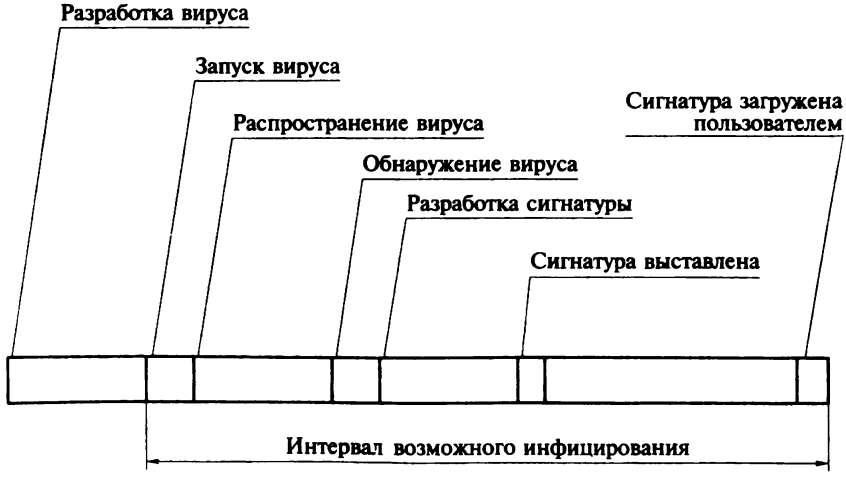


Рис. 2.7. Интервал возможного инфицирования пользователей

Антивирусные средства обеспечивают защиту от уже известных компьютерных вирусов, их вариантов и даже от некоторых новых вирусов. Проверка программных средств на наличие компьютерных вирусов в общем случае включает в себя:

- поиск вирусоподобных фрагментов кодов в программном средстве;
- моделирование ситуаций, предположительно способных вызвать активизацию компьютерного вируса;
- анализ особенностей взаимодействия компонентов программного средства с окружающей операционной средой.

Для поиска и удаления известного антивирусной компании вируса используется понятие «сигнатура». Под *сигнатурой* понимается уникальная последовательность байтов, принадлежащая конкретному известному вирусу и не встречающаяся в других программах.

Приведем сигнатуры некоторых вирусов:

- Concept:

57573649 6y666563 746a7206 06646a02 6904734в 65240c67

- Nuke-b:

6f616464 6e0467c2 80673b80 0506076a 083a5061 794c6f61

- W32.CIH.1003:

66837e06 177405fe 464debee 015e10c6 464d80eb e5880888

Согласно ГОСТ Р 51188 — 98 различают программные и аппаратно-программные методы обнаружения компьютерных вирусов.

Программные методы включают в себя:

- сканирование;
- обнаружение изменений;
- эвристический анализ;
- резидентную защиту (резидентный «сторож»);
- вакцинирование программных средств.

Сканирование заключается в том, что специальная антивирусная программа, называемая сканером, последовательно просматривает проверяемые файлы в поиске сигнатур известных вирусов. Антивирусные программы-сканеры, которые могут удалить обнаруженные вирусы, обычно называются полифагами. Для эффективного использования антивирусных программ, реализующих метод сканирования, необходимо постоянно обновлять их базы данных сигнатур (обычно это выполняется автоматически).

Обнаружение изменений основано на использовании антивирусных программ-ревизоров, которые запоминают в специальных файлах образы главной загрузочной записи, загрузочных секторов логических дисков, параметры всех контролируемых файлов, а также информацию о структуре каталогов и номера плохих кластеров дисков. Кроме того, могут контролироваться дополнительные параметры, такие как объем оперативной памяти, количество подключенных дисков и т. п. Программы-ревизоры потенциально могут обнаружить любые компьютерные вирусы, даже те, которые ранее не были из-

вестны. Дополнительной возможностью программ-ревизоров является способность восстановления инфицированных файлов и загрузочных секторов путем использования ранее запомненной информации.

Эвристический анализ реализуется с помощью антивирусных программ, которые проверяют программы ОС, прикладные программы и загрузочные секторы дисков и дискет, пытаясь обнаружить в них код, характерный для компьютерного вируса. Например, эвристический анализатор может обнаружить, что в проверяемой программе присутствует код, устанавливающий резидентный модуль в памяти. Эвристический анализ позволяет обнаруживать ранее неизвестные вирусы.

Резидентная защита (*резидентный сторож*) — это антивирусная программа, которая постоянно находится в оперативной памяти компьютера и отслеживает заданные действия, выполняемые другими программами, которые считаются подозрительными и свидетельствуют о наличии компьютерного вируса. При обнаружении таких действий резидентный сторож, как правило, сообщает пользователю о наличии подозрительных действий, ожидая его ответа (разрешение или запрещение данных действий). Обычно резидентные сторожа автоматически проверяют все запускаемые программы на наличие сигнатур известных вирусов.

Вакцинирование устанавливает способ защиты конкретной программы от вируса, при котором к этой программе присоединяется специальный модуль контроля, следящий за ее целостностью. При этом проверяются контрольная сумма программы или какие-либо другие ее характеристики. Если вирус заражает вакцинированный файл, то модуль контроля обнаруживает изменение контрольной суммы файла и сообщает об этом пользователю.

Аппаратно-программные методы основаны на применении любого (любых) из программных методов зашиты программных средств от компьютерных вирусов с помощью специальных технических устройств.

Как правило, они реализуются с помощью специализированного устройства — контроллера, вставляемого в один из разъемов расширения компьютера, и специального ПО, управляющего работой этого контроллера и приводящего в исполнение один или несколько из программных методов. Наличие такого устройства дает возможность защитить главную загрузочную запись, загрузочные секторы, выполняемые файлы, файлы конфигурации и т.д. Кроме того, аппаратно-программные средства позволяют защитить компьютер от неквалифицированного пользователя, не давая ему удалить важную информацию, переформатировать диск, изменить файлы конфигурации.

Начиная с Windows XP, в ОС имеется специальная подсистема защиты файлов — Windows File Protection (WFP). Данная система обеспечивает контроль за громадным числом системных файлов (около

1 700), проверяя их целостность. Копии всех этих защищаемых файлов находятся в директории \Windows\System 32\Cache (администратор системы может сменить расположение этой папки). Кроме того, ОС содержит ряд утилит работы с WFP.

Утилита проверки файлов sfc.exe позволяет проверить корректность версий всех системных файлов. Сведения о файлах с некорректным номером версии заносятся в протокол. Подсистема WFP проверяет корректность системных файлов с помощью механизма подписи драйверов. Дистрибутив ОС содержит каталог хешей файлов, каждый из которых подписан компанией Microsoft. Эти хеши и подписи хранятся в директории \system32\CatRoot.

Для проверки подписей текущих системных файлов используется утилита SigVerif.exe.

Применение WFP дало возможность защитить Windows XP от изменения основных системных файлов ОС. С появлением Windows Vista и Windows 7 компания Microsoft перешла к технологии защиты ресурсов (Windows Resource Protection, WRP).

Все методы защиты можно сгруппировать по времени их использования для защиты.

1. До обнаружения зловредного ПО: усиление системы — установка всех последних патчей и изменений, удаление ненужных или неиспользуемых модулей, отслеживание появления новых угроз на основных сайтах по безопасности, периодическое резервное копирование основных программ и данных.

2. При обнаружении: использование антивирусных средств для контроля всей входящей информации, применение специальных средств обнаружения, сканирование компьютера с использованием ОС, загружаемой с CDROM, которая может быть отличной от сканируемой.

3. В ходе устранения результатов атаки: сохранение следов атаки для предоставления правоохранительным органам, восстановление поврежденных программ и данных.

2.10. Технология Black и Whitelisting

Традиционный подход к обнаружению вредоносных программ заключался в так называемой технологии Blacklisting, которая работает, блокируя известные угрозы на основе образцов (сигнатур) вредоносных программ. Антивирусные компании поддерживают списки (базы данных) сигнатур известных вирусов, которые загружаются в компьютеры пользователей. При появлении нового вируса антивирусная компания формирует новую сигнатуру для его обнаружения и помещает в базу данных, доступную для загрузки. Эта технология предусматривает автоматическую загрузку новых сигнатур пользователями.

Основные достоинства технологии Blacklisting состоят в том, что:

- обновление списка сигнатур выполняется автоматически;
- она позволяет обнаруживать и удалять вредоносные программы;
- обновление осуществляется на лету;
- обеспечивается защита против известных угроз.

Как и всякая другая, данная технология имеет недостатки:

- пользователи предоставляют контроль за функционированием своего компьютера и сети антивирусной компании;
- необходимо постоянное обновление списка сигнатур (увеличение объема памяти, сетевого обмена и т. п.);
- существует уязвимость к вариантам известных атак и атакам zero-day;
- сканирование требует значительных затрат процессорного времени.

Одним из усовершенствований технологии Blacklisting являются *эвристики* — приложения, использующие знания, полученные от экспертов при решении заданных проблем. Эвристические программы базируются на известных источниках, которые содержат образцы вредоносных программ, извлеченных описаниях для их обнаружения (например, как анализировать структуры программ и их поведение), а также других характеристиках, применяемых вместо поиска сигнатур.

Эвристический анализатор — это программа, которая анализирует код проверяемого объекта и по косвенным признакам определяет, является ли объект вредоносным. Работа анализатора, как правило, начинается с поиска в коде подозрительных признаков (команд), характерных для вредоносных программ. Этот метод называется статичным анализом. Например, многие вредоносные коды ищут исполняемые программы, открывают найденные файлы и изменяют их. Эвристический анализатор просматривает код приложения и, встретив подозрительную команду, увеличивает некий «счетчик подозрительности» для данного приложения. Если после просмотра всего кода значение счетчика превышает заданное пороговое значение, то объект признается подозрительным.

Преимуществом этого метода является простота реализации и высокая скорость работы, однако уровень обнаружения новых вредоносных кодов остается довольно низким, а вероятность ложных срабатываний достаточно большой. Поэтому в современных антивирусных средствах статический анализ используется в сочетании с динамическим. Идея такого комбинированного подхода состоит в том, чтобы до того как приложение будет запущено на компьютере пользователя, эмулировать его запуск в безопасном виртуальном окружении, которое называется также буфером эмуляции, или «песочницей». Динамический эвристический анализатор считывает часть кода приложения в буфер эмуляции антивируса и с помощью

специальных «трюков» эмулирует его исполнение. Если в процессе этого «псевдоисполнения» обнаруживаются какие-либо подозрительные действия, то объект признается вредоносным и его запуск на компьютере пользователя блокируется.

К опасному, или вредоносному, поведению ПО может относиться следующая активность:

- доступ к ресурсам системы (например, к реестру системы);
- самокопирование программы на сетевые ресурсы, в каталог автозапуска, в системный реестр с последующей рассылкой своих копий;
- перехват ввода данных с клавиатуры;
- скрытая установка драйверов;
- изменение ядра ОС;
- создание скрытых объектов и процессов с отрицательными значениями идентификаторов (PID);
- внедрение в другие процессы;
- установление сетевых соединений и передача данных и т. п.

Например, антивирус Касперского (Kaspersky Internet Security 2011) использует статический набор и поддержку обновляемых эвристик. Кроме того, данный продукт поддерживает пробные поведенческие шаблоны. Если модуль проактивной защиты продукта обнаруживает поведение приложения как подозрительное, то в «Лабораторию Касперского» отправляется специальный отчет через сеть Kaspersky Security Network (KSN). Такая возможность реализуется, если пользователь подтвердил согласие на участие в KSN.

В отличие от статического метода динамический более требователен к ресурсам компьютера, так как для анализа необходимо использовать безопасное виртуальное пространство, а запуск приложения на компьютере пользователя откладывается на время анализа. Однако и уровень обнаружения вредителей у динамического метода значительно выше статического, а вероятность ложных срабатываний существенно меньше. Поэтому в последние годы широко стала использоваться новая технология Whitelisting, которая предлагает другой подход к обнаружению вредоносных программ. Ее основные черты — формирование списка легитимных программ (приложений) и защита их от изменения и запуска недоверенными программами. Это позволяет обеспечить защиту не только от вредоносных программ (вирусов и червей), но и от шпионского ПО и других недоверенных приложений.

Основными достоинствами технологии Whitelisting являются:

- отсутствие необходимости постоянного обновления, а также постоянного сканирования входящего и исходящего трафика (снижение нагрузки процессора);
- защита от атак zero-day;
- защита (блокировка) загрузки и инсталлирования недоверенных программ;

- блокирование запуска неавторизованных программ.

Большинство продуктов, использующих технологию Whitelisting, разрешают или запрещают выполнение программ на основе списка доверенных пользователей, доверенных путей поступления программы и доверенных производителей с помощью цифровых сертификатов. Некоторые ведут громадные списки известных файлов программ и приложений, для которых вычислены хеш-значения.

Например, компания Bit9 поддерживает громадный каталог программ для ОС Windows, UNIX и Linux, который, в частности, содержит следующие данные: имя и размер файла, дата и время создания файла, принадлежность файла к продукту, источники данных и уровень значимости файла.

«Лаборатория Касперского» в настоящее время предлагает технологию защиты на основе динамических списков доверенного ПО (Dynamic Whitelist, DW), которая содержит две составляющие: обновляемая категория файлов ОС (golden images) и процедура безопасных обновлений доверенными модулями обновления (trusted updaters).

Первая составляющая обеспечивает то, что включение защиты на основе DW не приведет к невозможности загрузки компьютеров из-за блокировки важного приложения на старте ОС. Компания поддерживает списки ключевых файлов для 18 локализаций всех основных редакций ОС Windows. При этом в этих списках присутствует не только ПО Microsoft, но и другие важные файлы, такие, например, как модули обновления драйверов устройств.

Вторая составляющая позволяет обновлять уже установленное на компьютере ПО безопасным способом. Компания разработала безопасную процедуру обновления продуктов, учитывающую сложные цепочки программных вызовов, выполняемых при этом.

Поскольку изменчивость является общей характеристикой доверенного и вредоносного ПО, то компания ведет базу знаний о «всем многообразии существующего программного обеспечения»: на данный момент число записей в «белой» базе компании «Лаборатория Касперского» превышает 230 млн.

Современные продукты компании «Лаборатория Касперского» реализуют и «облачные» технологии в виде KSN — инфраструктуры онлайн-служб и сервисов, позволяющей антивирусным средствам пользователей получать доступ к базам знаний Лаборатории в режиме реального времени. В работе KSN используются базы, содержащие информацию о доверенных (whitelisting) и подозрительных (Urgent Detection System, UDS) программах и Web-сайтах. Применяется также технология Wisdom of the Crowd (WoC), предоставляющая информацию о степени популярности программы и ее репутации среди пользователей KSN. Фактически данная инфраструктура представляет собой экспертную систему, направленную на анализ угроз безопасности компьютера. Через нее пользователи могут оперативно обмениваться между собой информацией об обнаруженных угрозах

и их источниках. В результате повышается скорость реакции антивируса на новые виды угроз, увеличивается эффективность работы некоторых компонентов защиты, а также снижается вероятность ложных срабатываний.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое вредоносная программа?
2. Какие основные функции способна выполнять вредоносная программа?
3. Перечислите основные виды вредоносных программ.
4. Что понимается под программной закладкой?
5. Дайте определение компьютерного вируса.
6. Назовите основные параметры классификации компьютерных вирусов.
7. Перечислите основные виды файловых вирусов.
8. В чем состоят основные особенности макровирусов?
9. Укажите основные характеристики загрузочных вирусов.
10. Перечислите основные методы защиты компьютерных вирусов от обнаружения.
11. Дайте определения полиформизма и метаморфизма компьютерных вирусов.
12. Что такое троянский конь (троян)?
13. Какие основные виды троянских коней вам известны?
14. Дайте определение сетевого червя.
15. Перечислите основные методы защиты от обнаружения, применявшиеся червем Морриса.
16. Какие основные механизмы содержит сетевой червь?
17. Что такое потайной ход?
18. Укажите основные виды руткитов.
19. Перечислите основные функции ядра ОС.
20. Назовите элементы наименований вирусов.
21. Приведите доказательство неразрешимости проблемы обнаружения вирусов.
22. По каким признакам можно судить об инфицировании компьютерным вирусом?
23. Укажите основные методы обнаружения компьютерных вирусов.
24. Проведите сравнение технологий Blacklisting и Whitelisting.

Глава 3
УДАЛЕННЫЕ СЕТЕВЫЕ АТАКИ

— Привяжи покрепче, а то отрежет мне ненароком голову, — сказал Траляля. И, подумав, мрачно прибавил: — Знаешь, одна из самых серьезных потерь в битве — это потеря головы.

Л. Кэрролл. Алиса в стране чудес

Основные виды угроз были рассмотрены в гл. 1 и 2. Далее рассмотрим угрозы при сетевом взаимодействии.

Общепринято выделять следующие основные угрозы:

- угрозы целостности;
- угрозы конфиденциальности;
- угрозы доступности.

Эти обобщенные виды угроз не дают представления о конкретной угрозе. Поэтому, исходя из общей схемы межсетевого взаимодействия, для случая удаленных атак можно выделить два основных типа угроз.

1. Угрозы, вызываемые участниками информационного обмена:
   - отказ от получения данных после их получения;
   - отказ от передачи данных после их передачи;
   - отказ от достигнутого соглашения.
2. Угрозы, вызываемые третьей стороной (атакующим):
   - вставка данных в обмен;
   - отказ в обслуживании.

Среди угроз для сети организации и ее систем можно выделить старые и новые угрозы.

Старые угрозы реализуются атаками, базирующимися на использовании хорошо известных уязвимостей и скриптов атак (эксплойтов). Такие угрозы исходят от недостаточно компетентных хакеров (называемых script kiddies) или совершенно некомпетентных (называемых newbies). Эти категории нарушителей используют готовые скрипты атак и могут совершенно не понимать действительных механизмов применяемых (используемых) эксплойтов, а также их возможных побочных действий. Но это не уменьшает их опасность для организаций, так как реализация старых незащищенных угроз может нанести значительный ущерб, если организация не примет соответствующих мер.

Новые угрозы являются более серьезными и потенциально опасными для организации. Эти угрозы характеризуются направлен-

ными попытками нанести ущерб, получить информацию, нарушить операции функционирования и т.д. Реализуют новые угрозы обычно квалифицированные взломщики, обладающие детальными знаниями механизмов сетевого взаимодействия и логики функционирования приложений. Для получения необходимой информации нарушители используют специально разработанные средства и скрипты (которые потом могут использовать более слабые категории нарушителей для проведения своих атак). Как правило, новые угрозы используют неизвестные или только что обнаруженные уязвимости.

Все множество угроз можно разделить на внешние и внутренние. Внешними угрозами являются те, которые исходят извне. Внутренние угрозы инициируются субъектом, имеющим доступ к инфраструктуре организации. Классическим примером внутренней угрозы является случай, когда обиженный увольнением сотрудник наносит ущерб информации организации.

3.1. Сетевые атаки

Каждый год открываются новые уязвимости, но знания, необходимые для проведения атаки, уменьшаются, чему в значительной мере способствует Интернет (рис. 3.1).

Новые и старые внешние угрозы реализуются посредством сетевых атак или удаленных сетевых атак. Под удаленной сетевой атакой будем понимать воздействие на программные компоненты целевой системы с помощью программных средств. Таким образом, атака является попыткой получить данные или осуществить проникновение. Обычно выделяют три основных типа атак:

1) атаки разведки (проб, сбора информации);
2) атаки получения доступа;
3) атаки отказа в обслуживании.

Эти типы атак не всегда используются отдельно и обычно применяются в сочетании для достижения атакующим своих целей.

Атаки разведки используются для сбора информации о целевой сети или системе. Такие атаки кажутся безобидными для целевой системы и могут рассматриваться сетевыми администраторами как «сетевой шум» или надоедливое поведение. Но информация, собранная на этапе разведки, используется для проведения атаки. Средства проведения разведки могут быть как обычными, входящими в состав ОС, так и специально разработанными. Поскольку точные знания о целевой системе и ее уязвимостях могут обеспечить успешность атаки, атаки разведки должны рассматриваться как серьезная угроза.

Атаками получения доступа являются такие атаки, которые включают неавторизованное использование целевого хоста или

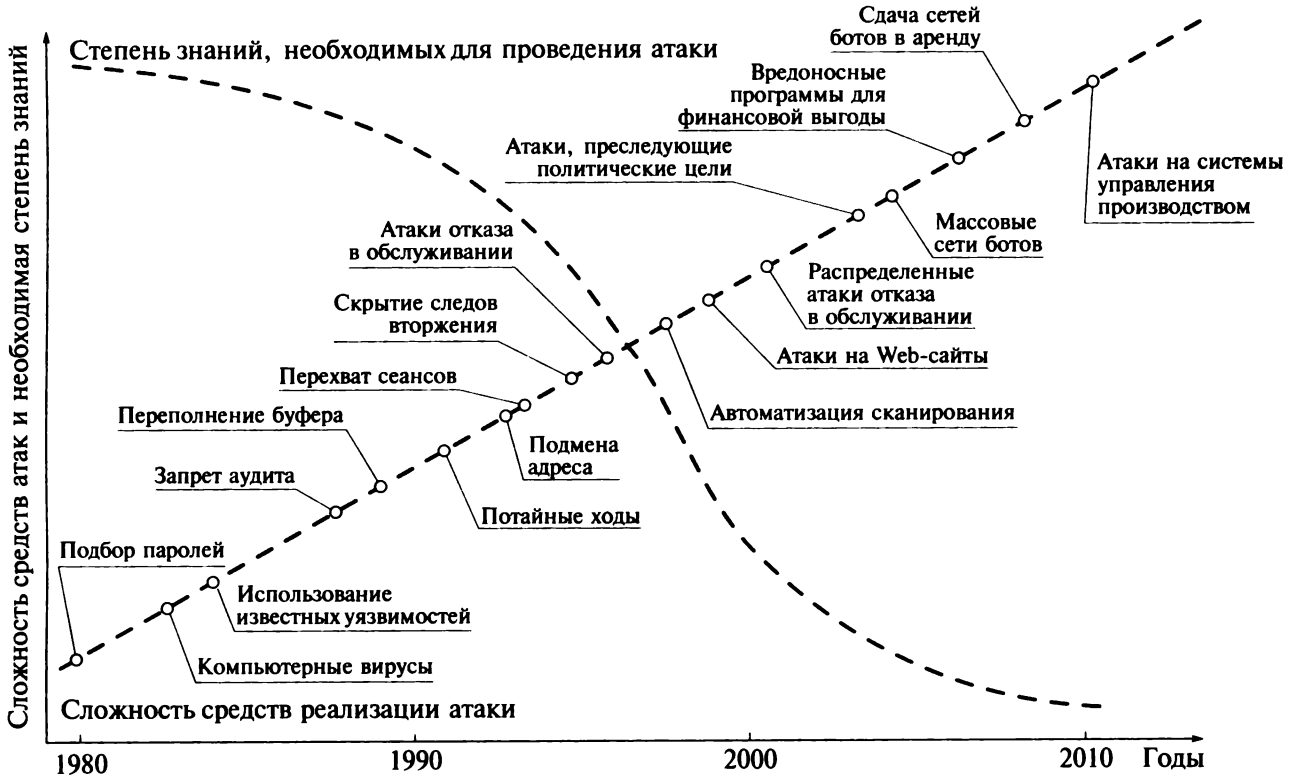


Рис. 3.1. Сложность средств атак и необходимая степень знаний

группы хостов. Средство, с помощью которого атакующий получает доступ к инфраструктуре, обычно зависит от используемой уязвимости, которая присутствует в ОС, в приложении или в защитном механизме. Часто эти уязвимости открываются атакующим при проведении разведки. Атаки получения доступа могут осуществляться вручную или с использованием автоматизированных или даже автоматических средств.

Атаки получения доступа можно разбить на три вида неавторизованной деятельности:

- извлечение данных (чтение, копирование, перемещение);
- доступ к системе (нарушитель получает реальный доступ к системе с различным уровнем привилегий);
- расширение привилегий (необходимо атакующему как для полного управления системой, так и для скрытия своего взлома).

Третьим типом атак являются *атаки отказа в обслуживании*, когда атакующий пытается препятствовать доступу легальных пользователей к системе или службе. Часто эти атаки реализуются переполнением ресурсов инфраструктуры запросами (легитимными или поддельными) на доступ к службе. Такие атаки могут быть направлены как на отдельный хост, так и на сеть в целом.

Одной из серьезных проблем в области компьютерной безопасности является отсутствие единой терминологии. Данная проблема усугубляется следующими обстоятельствами:

- многообразием используемых терминов, которые уже существуют в языке;
- преобладанием переводных книг, в которых переводчики используют неоднозначные термины (исключением из этого правила является блестящий перевод книги «Новый словарь хакера», в котором, к сожалению, не содержатся термины, вошедшие в компьютерный обиход за последние годы);
- некорректным использованием производителями и продавцами средств защиты терминов, которые должны убедить покупателя приобретать именно их продукт;
- отсутствием стандартизованных списков терминов и устоявшейся терминологии.

Любая сетевая атака направлена на программное средство атакуемого хоста. В качестве атакуемого программного средства может выступать сетевой стек операционной системы, другой системный код, прикладная программа, т.е. элемент прикладного или системного программного обеспечения. Атака, как правило, возможна из-за наличия ошибок и просчетов при разработке, реализации, настройке или использовании данного программного средства.

Рассмотрим основные элементы терминологии сетевых атак.

Ошибка — погрешность в программном коде данного программного средства. Возможны ошибки, которые не проявились или еще не были использованы злоумышленниками.

Просчет — недостаток программного средства, который определяется как его программным кодом, так и недостатком самого проекта или способом применения средства.

Ошибки и просчеты представляют собой уязвимости.

Уязвимость — это недостаток программного средства, которым может воспользоваться злоумышленник.

Злоумышленник для известной ему уязвимости разрабатывает или использует готовые (разработанные другими) шаблоны атак. Экземпляр шаблона атаки, созданный для компрометации конкретного фрагмента кода программного средства, является программой атаки, или эксплойтом.

При проведении атаки злоумышленник использует сценарий атаки, который предусматривает использование различных шаблонов в зависимости от поведения (реакции) атакуемой системы. Таким образом, атака — это процесс реализации некоторого сценария атаки. В ходе атаки злоумышленник получает данные (реакции атакуемой системы), которые свидетельствуют об успехе (неудаче) применения данного шаблона или служат основанием для применения определенного шаблона атаки. Описание каждой атаки может быть основано на используемых ею уязвимостях атакуемой системы.

Успешная атака называется *вторжением*. При осуществлении вторжения злоумышленник достигает своей основной цели — получает доступ к системе, приобретает возможность исполнения своего программного кода или вызывает прекращение (ограничение) выполнения функций атакованной системы. Дальнейшие цели или этапы действий злоумышленника могут включать в себя расширение полученных привилегий, внедрение своего программного кода, принятие мер по маскировке своего присутствия и факта вторжения и т. д.

3.2. Обобщенный сценарий атаки

Статистика нарушений безопасности показывает, что количество атак имеет тенденцию к экспоненциальному росту. Сама сеть Интернет является благодатной почвой для вторжений в компьютерные системы. Объединение компьютеров в сети позволяет пользователям совместно использовать данные, программы и вычислительные ресурсы. Кроме того, громадное число эксплойтов доступно в Интернете. Поэтому даже пользователи с минимальными познаниями могут осуществить успешный взлом. Это связано с тем, что значительная часть пользователей Интернета не уделяет достаточного внимания проблемам обеспечения безопасности. При обнаружении уязвимости в программном продукте требуется время для ее устранения. Это время складывается из времени разработки корректирующей про-

граммы (заплатки, патча — patch), установки этого патча на соответствующий сервер компании и выставления объявления о наличии патча. Это требует от пользователя или системного администратора постоянного просмотра соответствующих сайтов производителей программного обеспечения и программных продуктов. Далее требуется установка соответствующего патча на компьютер. При наличии в организации множества компьютерных систем, множества операционных систем и программных продуктов такие операции становятся достаточно дорогими и ресурсоемкими. Поэтому значительная часть пользователей и не подозревает о наличии уязвимостей, наличии соответствующих патчей и необходимости их установки. В таком случае злоумышленнику нужно только найти соответствующую компьютерную систему.

Рассмотрим обобщенный сценарий атаки, который можно представить в виде следующих шагов:

- пассивная разведка;
- активная разведка;
- выбор (разработка) эксплойта;
- взлом целевой системы;
- загрузка «полезного груза» (которым, как правило, является вредоносная программа);
- сокрытие следов взлома.

Конечно, данная последовательность может быть нарушена или могут быть исключены отдельные шаги данного сценария. Кратко рассмотрим эти этапы.

3.2.1. Пассивная разведка

Данный этап называется пассивным, так как злоумышленник не входит в непосредственный контакт с целью или входит с соблюдением общепринятых правил (например, посещая сайт компании, куда входит целевая система).

Целью данного этапа является сбор информации о цели, поэтому часто его называют рекогносцировкой цели. В качестве цели может выступать как конкретный хост (сервер), так и целая сеть организации. Достаточно часто цель выбирается из условия простого поиска системы, содержащей известную уязвимость, к которой злоумышленник имеет эксплойт.

Для сбора информации могут использоваться существующие в Интернете сайты и базы данных, предоставляющие сетевые адреса доменных имен и блоки сетевых адресов. Злоумышленник может отыскивать номера телефонов, имена и фамилии персонала организации, их почтовые адреса. Большой интерес для злоумышленника представляют партнерские и дочерние организации, взаимодействующие с целевой организацией. Кроме того, используются всевоз-

можные открытые источники и публикации (конференции, службы передачи сообщений, рекламные материалы и т.д.).

Сбору информации способствуют определенные базы данных, находящиеся в Интернете, и специальные программы, позволяющие получить такую информацию. Среди них отметим whois, базу данных ARIN (American Registry for Internet Numbers), APNIC (Asia-Pasific Network Information Center) и др.

Например, запрос в whois, содержащий название компании Microsoft, дает следующие результаты (значительная часть информации опущена для краткости):

```
   Microsoft Corp (MSFT)
   Microsoft Corporation (ZM23-ARIN) noc@microsoft.com +1-
425-882-8080
   Microsoft (ZM39-ARIN) noc@microsoft.com +1-425-882-8080
   Microsoft Corp (AS13811) MSLI 13811
   Microsoft Corp (AS14719) MICROSOFT-CORP-BCENTRAL 14719
   Microsoft Corp (AS8068) MICROSOFT-CORP——MSN-AS-BLOCK
8068 - 8075
   Microsoft Corp (AS3598) MICROSOFT-CORP-AS 3598
   Microsoft Corp (AS5761) MICROSOFT-CORP——MSN-AS——SATURN
5761
   Microsoft Corp (AS6182) MICROSOFT-CORP—MSN-AS-4 6182
   Microsoft Corp (AS6194) MICROSOFT-CORP—MSN-AS-3 6194
   Microsoft Corp (AS6291) MICROSOFT-CORP——MSN-AS 6291
   Microsoft Corp (AS13399) MICROSOFT-CORP——MSN-AS-2 13399
   Microsoft Corp (AS23468) MICROSOFT-CORP-XBOX-ONLINE
23468
   Microsoft Corp NETBLK-MSOFT-NET (NET-198-105-232-0-1)
198.105.232.0 - 198.105.235.255
   Microsoft Corp MSOFT-2 (NET-198-105-233-0-1) 198.105.233.0
- 198.105.233.255
   Microsoft Corp MSOFT-4 (NET-198-105-235-0-1) 198.105.235.0
- 198.105.235.255
   Microsoft Corp MSOFT-3 (NET-198-105-234-0-1) 198.105.234.0
- 198.105.234.255
   Microsoft Corp MSOFT-1 (NET-198-105-232-0-2) 198.105.232.0
- 198.105.232.255
   Microsoft Corp MICROSOFT-1 (NET-199-103-90-0-1)
199.103.90.0 - 199.103.91.255
   Microsoft Corp MICROSOFT-CORP-MSN-3 (NET-199-103-122-0-1)
199.103.122.0 - 199.103.122.255
   Microsoft Corp MICROSOFT17 (NET-199-6-92-0-1) 199.6.92.0
- 199.6.94.255
   Microsoft Corp MICROSOFT-2 (NET-204-79-7-0-1) 204.79.7.0
- 204.79.7.255
```

Microsoft Corp MICROSOFT-NET1 (NET-204-79-27-0-1)
204.79.27.0 – 204.79.27.255 и т.д.

3.2.2. Активная разведка

Этап проведения активной разведки называют сканированием. Злоумышленник пытается определить структуру интересующей его сети, точки доступа, доступные узлы и хосты, расположение маршрутизаторов и межсетевых экранов, установленные операционные системы и их версии, открытые порты и работающие службы, версии установленных программных продуктов. На этом этапе злоумышленник входит в контакт с объектами интересующей его системы, поэтому его действия могут быть обнаружены средствами защиты целевой системы.

Для решения задач проведения активной разведки (сканирования) злоумышленник может использовать большое количество разнообразных средств, многие из которых являются общедоступными. Среди них такие средства, как ping, traceroute, nmap (http://www.insecure.org/nmap), SuperScan (http://www.foundstone.com). Свободное наличие таких средств во многом обусловлено тем, что они активно используются системными администраторами для решения текущих проблем управления и защиты сетей.

Существует множество методик и средств, позволяющих определить операционную систему, установленную на хосте, и ее версию. Разработаны методы так называемого скрытого (stealth) сканирования, не позволяющие определить источник, проводящий сканирование. Кроме того, может использоваться «замедленное» сканирование, когда злоумышленник посылает отдельные запросы через большие интервалы времени. В этом случае достаточно высока вероятность того, что на целевой системе не обратят внимания на отдельные запросы.

3.2.3. Выбор эксплойта

На основании полученных данных злоумышленник выбирает (модифицирует имеющийся или разрабатывает новый) эксплойт для проведения взлома.

Большое число малоопытных взломщиков начинают взлом на основании имеющегося эксплойта. Тогда на этапе разведки ищется система, которая имеет соответствующие уязвимости. При этом возможен простой перебор адресов, чтобы каким-то образом (например, случайно) выбрать уязвимый хост, который и станет целью взлома. Напомним, что IP-адрес в пакете занимает 32 разряда и представляет собой двоичное число. Имена же хостов, которые задаются

в браузере, имеют вид, удобный для пользователя. Например, можно задать адрес компании Microsoft в виде www.microsoft.com или 207.46.20.30. Этот же адрес можно представить в шестнадцатеричной системе счисления (CF2E141E) или двоичной (11001111001011100001 010000011110). Поэтому достаточно легко использовать в эксплойте программный цикл перебора значений адреса. Данный метод был использован в январе 2003 г. червем Slammer, который для перебора адресов использовал следующую рекуррентную зависимость: $x_{n+1} = (214\,013x_n + 2\,531\,011) \bmod 2^{32}$. Применение данного метода ограничивается тем обстоятельством, что во всем диапазоне адресов имеются неиспользуемые адреса.

После получения необходимой информации и выбора эксплойта злоумышленник может переходить к взлому системы.

3.2.4. Взлом целевой системы

Существует множество различных способов взлома. К их числу можно отнести получение доступа к системе, расширение имеющихся или полученных полномочий и отказ в обслуживании.

Взлом практически всегда выполняется с помощью программного обеспечения и направлен на программное обеспечение.

Успешная атака включает в себя несколько последовательных действий. Напомним, что шаблон атаки — это вариант взлома по отношению к уязвимому месту программного обеспечения. Поэтому в шаблоне атаки может быть предусмотрено использование нескольких свойств уязвимых мест и должны быть указаны данные, необходимые для взлома системы. Среди методов взлома можно выделить такие, как внедрение команд, использование каналов и портов, изменение прав доступа, использование свойств файловой системы, манипулирование переменными среды, использование внешних переменных, использование некорректных данных, подбор параметров, использование некорректной обработки ошибок и т. д.

Методы взлома могут простыми и сложными. В качестве простого взлома (уже устраненного во всех системах) приведем пример посылки пакета ping, общий размер которого превышает допустимую величину:

```
C:\> ping -l 65550 www.target.com
```

Сейчас ОС Windows на такую команду выдает сообщение:

```
Недопустимое значение параметра -l, допустимый диапазон с 0 по 65500.
```

Атака уже упоминавшегося червя Slammer состояла в посылке всего одного пакета UDP размером в 404 байт. Собственно червь занимал 376 байт и использовал уязвимость Microsoft SQL Server или

MSDE 2000, открытую еще в июле 2002 г. Несмотря на это за первые 30 мин распространения червь инфицировал около 75 000 хостов.

Нападающий и защищающийся находятся в неравных условиях. Для защиты системы от атак необходимо знание обо всех возможных атаках против данной системы, а для проведения атаки достаточно найти только одну эффективную программу атаки.

3.2.5. Загрузка «полезного груза»

Как правило, атака проводится не только ради самого процесса взлома. Обычно злоумышленник хочет иметь возможность возврата во взломанную систему или использования ее в качестве плацдарма для атаки других систем. Кроме того, его может интересовать получение конфиденциальных данных во взломанной системе. К действиям по загрузке «полезного груза» атаки можно отнести следующие:

- установка программ «прослушивания» сетевого трафика локальной сети, в которой находится взломанная система, для получения информации, позволяющей осуществить взлом сетевых соседей;
- осуществление перенаправления портов, чтобы обеспечить передачу пакетов на взломанную систему, минуя межсетевой экран;
- установка «заплаток» на использованную или имеющиеся уязвимости, чтобы исключить взлом системы другими злоумышленниками;
- создание фальшивых учетных записей, наделенных привилегиями суперпользователя;
- создание или установка готового потайного входа (backdoor) во взломанную систему;
- установка «троянских коней» для сбора конфиденциальной информации, последующего входа в систему или для проведения атак других систем.

Эти и другие операции проводятся при отключении аудита на взломанной системе.

3.2.6. Сокрытие следов взлома

Чтобы администратор или пользователь взломанной системы не смог обнаружить наличие следов взлома, модификации и вставки в программное обеспечение, злоумышленник прибегает к удалению следов своего пребывания во взломанной системе. Для этого используются программы удаления записей из различных журналов, ведущихся в системе, скрытие установленных файлов, использование потоков файлов в файловой системе NTFS операционной системы Windows, троянизирование системных программ и утилит и замена программных компонент операционной системы.

Для реализации этих задач используются специальные наборы средств (rootkit), которые часто называют наборами средств для взлома.

3.3. Атаки «отказ в обслуживании»

Атаки, предназначенные для блокирования доступности компьютерных систем, называют *атаками «отказ в обслуживании»* (Denial of Service, DoS). Под *доступностью* понимается способность пользователей использовать желаемый ресурс или информацию надежным и постоянным способом. *Отказ в обслуживании* — это угроза, которая потенциально может нарушить доступность ресурсов системы, поэтому целью такой атаки является действие (или множество действий), выполняемое злоумышленником или программой, направленное на то, чтобы сделать ресурс недоступным для законных пользователей.

CERT Coordination Center определяет три основных типа таких атак:

1) исчерпание ограниченных или невозобновляемых ресурсов;

2) разрушение или изменение конфигурационной информации;

3) физическое разрушение или преобразование сетевых компонентов.

В 1996 г. появились первые сетевые атаки, которые использовали недостатки реализаций стека TCP/IP, в основном неопределенности в описании основных протоколов стека TCP/IP в соответствующих RFC (Request for Comments). Эти неопределенности в свою очередь определили различные варианты реализации стеков в разных ОС и привели к появлению различных атак. Позже возникли атаки, когда машины жертвы (сайты) заполнялись множеством запросов на соединение, исходящих из большого количества источников. В результате происходило «затопление» сайтов легитимными запросами, которые они физически не могли выполнить (данный метод назвали «затоплением» от термина flood).

Уже в 1998 г. появились первые средства реализации атак «отказ в обслуживании» и, начиная с 1999 г., эти атаки стали постоянными.

Первоначально для проведения атак «отказ в обслуживании» злоумышленники использовали свой компьютер с подельными адресами источника, а позднее — инфицированные злоумышленником компьютеры. Для установки средств проведения атак (генераторов атак) злоумышленник сначала должен был осуществить взлом целевой системы и затем установить на ней потайной ход, содержащий средство удаленных атак. Конечно, для сокрытия присутствия в инфицированной системе применялись руткиты.

Развитию атак «отказ в обслуживании» способствовало широкое распространение IRC (Internet Relay Chat, ретранслируемый

интернет-чат) — сервисной системы, с помощью которой можно общаться с другими людьми в режиме реального времени (создан финским студентом Ярко Ойкариненом в 1988 г.).

В начале 90-х годов XX в. пользователи IRC создали средства автоматизации некоторых задач и для защиты от некоторых атак (net split). Позже появились средства для удаления каналов (kill channels, chat rooms) и удаления пользователей из этих каналов так, чтобы они не могли вернуться. Такие средства получили название «бот» (bot) — сокращение от software robot. *Бот* — это программное обеспечение, которое позволяет удаленно контролировать и управлять инфицированной системой и может также использоваться для автоматизации определенных задач в IRC. Эти средства позволили инфицировать компьютеры пользователей и использовать их для проведения атак «отказ в обслуживании». В этом случае на целевую систему посылались атаки из громадного множества инфицированных источников. Причем реальные пользователи инфицированных систем и не подозревали о том, что являются источниками атак.

В настоящее время под ботами понимаются программные средства, которые функционируют автономно и автоматически. Инфицированный компьютер иногда называют «зомби» (zombie). Злоумышленники создают целые сети из ботов, которые называются *сетями ботов* (botnets).

Злоумышленник, управляющий сетью ботов, называется *пастухом* (bot herder). Схема построения IRC-сети ботов приведена на рис. 3.2.

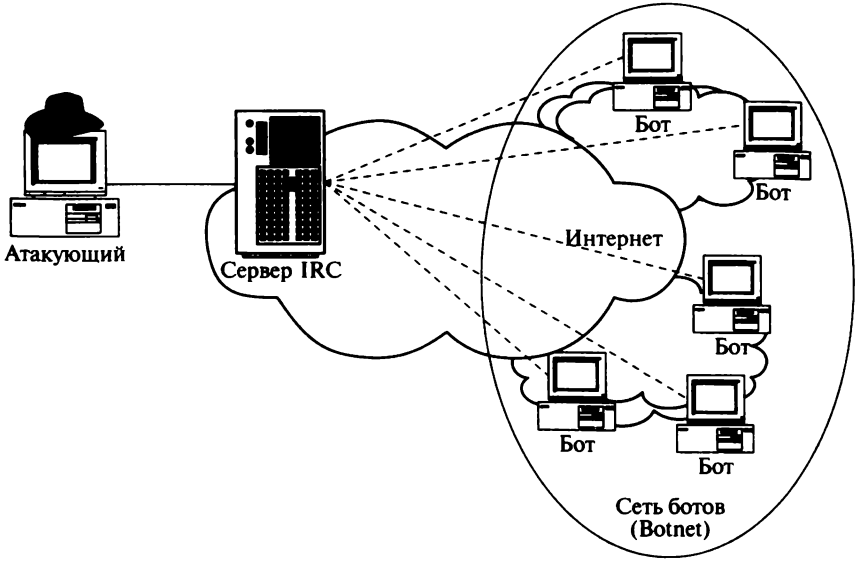


Рис. 3.2. Схема организации сети ботов

Первые примеры появления зловредных ботов относятся к 1999 г., когда Sub7 и Pretty Park ввели концепцию того, что инфицированная машина соединялась с IRC-каналом, слушая команды.

В 2000 г. появился бот Global Threat bot (GTbot), использовавший клиент-mIRC, который позволял запускать некоторые скрипты в ответ на события IRC и имел доступ к сокетам TCP и UDP.

В 2002 г. произошла эволюция в технологии сетей ботов, представителями которой явились SDBot и Agobot. SDBot имел небольшой исполняемый файл, написанный на C++, и распространялся с исходным кодом, что привело как к появлению множества его вариантов, так и расширению его идей. Agobot ввел концепцию модульного исполнения, в одном из модулей содержался модуль проведения атак «отказ в обслуживании». При инфицировании машины инсталлировался ПХ, после чего осуществлялась попытка отключить антивирусное ПО и блокировался доступ к сайтам производителей безопасности. Уже к концу 2002 г. было обнаружено более 580 вариантов Agobot. Например, на июнь 2005 г. были зафиксированы функционирующие сети ботов: Sdbot (12 800 вариантов), Agobot (3 821), Spybot (2 116), Polybot (106) и Mytob (228).

В 2003 г. появились первые боты для работы в P2P (бот Sinit).

Постепенно боты мигрировали от начального IRC Command & Control канала к коммуникации посредством HTTP-, ICMP- и SSL-портов, часто используя собственные протоколы.

Около 2003 г. криминальные структуры стали проявлять интерес к ботам с точки зрения организации спама. Примерами могут служить боты Bagle, Bobax и Mytob. Боты Bagle и Bobax были первыми сетями ботов для распространения спама. Mytob представлял собой смесь MyDoom (mass mailing worm) и SDBot. Это позволило криминальным элементам построить громадные сети ботов и распространять спам через компьютеры пользователей.

Подобные сети ботов используются для проведения распределенных атак «отказ в обслуживании», рассылки спама, сбора персональных данных пользователей инфицированных машин, хранения

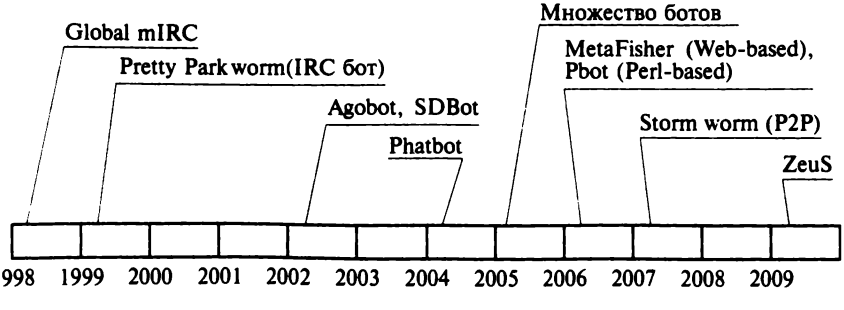


Рис. 3.3. Диаграмма появления сетей ботов

нелегального ПО и т. д. О количестве ботов в создаваемых злоумышленниками сетях, предназначенных для проведения распределенных атак «отказ в обслуживании» и рассылки спама, можно судить по следующим данным (2007 г.): сеть Storm содержала 230 000 агентов, Rbot — 40 000, Bobax — 24 000.

В настоящее время количество ботов в таких сетях измеряется сотнями тысяч и даже миллионов.

При рассмотрении сетей ботов обычно выделяют следующие основные параметры:

- модель контроля и управления (Command and Control, C&C) (централизованная, децентрализованная, сети P2P);
- используемые протоколы (IRC, IM, P2P, HTTP);
- механизм распространения («встроенные» IP-адреса, использование динамической DNS и т. д.).

Хронология появления сетей ботов приведена на рис. 3.3.

Сети ботов в настоящее время стали одним из самых прибыльных источников дохода в Интернете. Этому способствует, с одной стороны, хищение конфиденциальных данных из инфицированных компьютеров, а с другой — продажа и аренда злоумышленниками построенных ими сетей. Сети ботов могут использоваться злоумышленниками для решения и чисто криминальных задач разного масштаба: рассылка спама, кибершантаж и атаки на государственные сайты и сети.

3.3.1. Распределенные атаки «отказ в обслуживании»

Атаки, исходящие из одного источника, можно было достаточно легко фильтровать или блокировать. Кроме того, наличие адреса источника позволяло определить местоположение атакующего. Поэтому злоумышленники перешли к технологии проведения распределенных атак, когда целевая система атаковалось значительным числом машин.

Первые средства проведения таких атак появились в 1998 г. Уже в 1999 г. CERT опубликовал предупреждение Incident Note 99-04, в котором отмечалось широкое распространение вторжений на базе службы Solaris RPC. В этом же году были проведены первые атаки.

Переход к многочисленным источникам атаки, направленным на конкретную цель, показал силу подобных атак. Например, хакер Mafiaboy в 2000 г. практически парализовал работу нескольких публичных сайтов, таких как Yahoo!, eBay, Amazon и CNN.

Программные средства, реализующие подобные атаки, называются *средствами распределенных атак «отказ в обслуживании»* (Distributed Denial of Service, DDoS). Схема распределенной атаки «отказ в обслуживании» приведена на рис. 3.4.

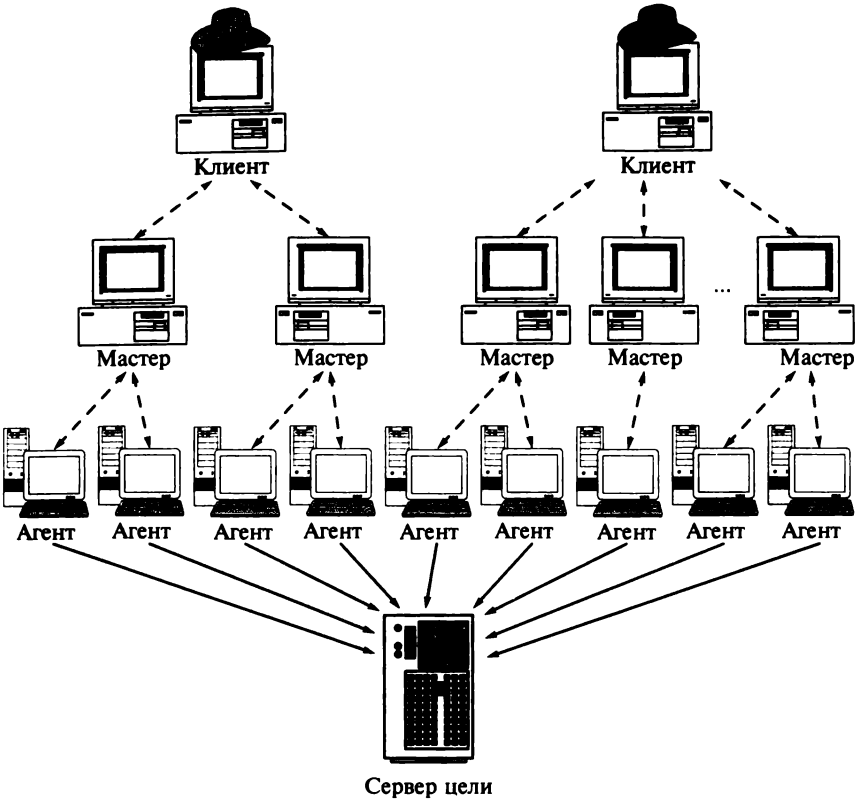


Рис. 3.4. Схема распределенной атаки «отказ в обслуживании»

Главным в атаке DDoS является хост, обычно называемый *клиентом* (client) и используемый тем, кто координирует атаку. Ниже уровнем находится хост (или хосты), называемый *хозяином* (master) или *обработчиком* (handler). Хозяин управляет подчиненными хостами, он организует атаки. На нижнем уровне расположены хосты,

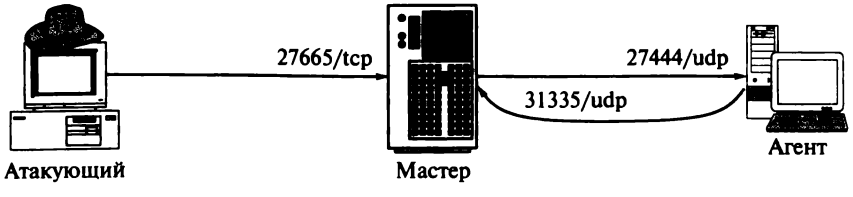


Рис. 3.5. Схема управления сетью Trin00

Таблица 3.1. **Некоторые команды атакующего мастеру**

Имя команды	Значение
`mtimer N`	Установить продолжительность атаки в *N* с (1 < *N* < 1 999). По умолчанию *N* = 300
`dos IP`	Адрес атакуемой цели
`mdos <ip1:ip2:ip3>`	Задание адресов для проведения атаки на множество хостов

называемые *агентами* (agents) или *демонами* (daemons). Именно агенты начинают атаки.

В среде злоумышленников и в открытом доступе стали появляться программные средства DDoS: Trin00, Tribe Flood Network (TFN), TFN2K, Stacheldraht, Mstream и Shaft.

Рассмотрим схемы организации некоторых атак.

Атака Trin00 использует для связи компонентов сети протоколы TCP и UDP (рис. 3.5). Все соединения (для передачи данных и команд) осуществляются при предъявлении пароля. Перечень некоторых команд для мастера (master, handler) приведен в табл. 3.1.

Некоторые команды, выдаваемые мастером агентам (daemons), приведены в табл. 3.2. Как видно из этих таблиц, команды предоставляют злоумышленнику значительные возможности по управлению зловредной сетью и нанесению ущерба.

Таблица 3.2. **Некоторые команды мастера агентам**

Команда	Содержание команды
`aaa pass IP`	Атака заданного адреса посылкой UDP-пакета на случайные номера портов (0 — 65 534) в течение заданного промежутка времени. По умолчанию — 120 с или заданное командой *bbb*. Размер посылаемого пакета задается командой *rsz*, по умолчанию — 1 000 байт
`bbb pass N`	Задание продолжительности атаки
`dle pass`	Отключение демона trinoo
`rsz N`	Установка размера буфера для проведения атаки в *N* байт
`xyz pass 123:ip1:ip2:ip3`	Задание множества адресов целей

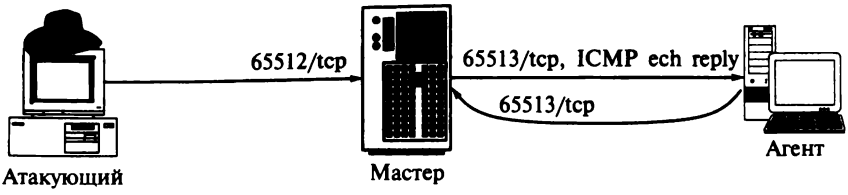


Рис. 3.6. Схема управления сетью Stacheldraht

Интересным примером другой зловредной сети является средство *Stacheldraht* (колючая проволока, нем.). Схема управления сетью приведена на рис. 3.6.

Это средство использует множество атак: ICMP-flood, SYN-flood, UDP-flood и Smurf, которые могут применяться отдельно или в совокупности. Для сокрытия передаваемых данных и команд используется алгоритм шифрования Blowfish. В программной реализации *Stacheldraht* имеется ограничение — один мастер может управлять не более чем 6 000 агентами. После предъявления пароля злоумышленник получает подсказку по командам, которые он может выполнять:

```
------------------------------------------------
stacheldraht(status: a!1 d!0)>.help
available commands in this version are:
------------------------------------------------
.mtimer .mudp .micmp .msyn .msort .mping
.madd .mlist .msadd .msrem .distro .help
.setusize .setisize .mdie .sprange .mstop .killall
.showdead .showalive
------------------------------------------------
```

Рассмотрим некоторые команды:

- `.mudp ip1[:ip2[:ipN]]` — начать атаку UDP flood на заданные адреса;
- `.micmp ip1[:ip2[:ipN]]` — начать атаку ICMP flood на заданные адреса;
- `.msyn ip1[:ip2[:ipN]]` —начать атаку SYN flood на заданные адреса;
- `.madd ip1[:ip2[:ipN]]]` — добавить в список атакуемых компьютеры с заданными адресами;
- `.mtimer` — задать продолжительность атаки (в секундах).

Другие команды предназначены для управления самой сетью.

3.3.2. Распределенные рефлекторные атаки «отказ в обслуживании»

При реализации атак «отказ в обслуживании» злоумышленнику нужно послать на целевую систему как можно больше пакетов. Для

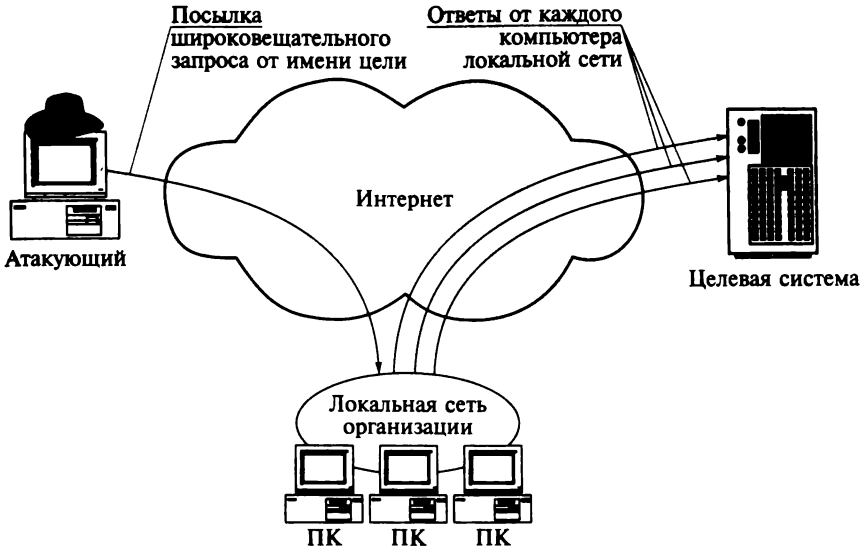


Рис. 3.7. Усиление атаки посылкой широковещательных запросов

увеличения их количества в первое время использовались так называемые *широковещательные запросы* (broadcast — циркулярные рассылки). Адрес циркулярной рассылки определяется как «последний» адрес в подсети или (в старых реализациях стека) как первый адрес в подсети. Такие адреса в простейшем случае оканчиваются на .0 и .255. Получив пакет с таким адресом, целевая система отправляет поток пакетов ICMP echo reply на указанный злоумышленником адрес жертвы. Такой метод называют *усилением* или *рефлектором* (reflector). Схема усиления атаки приведена на рис. 3.7.

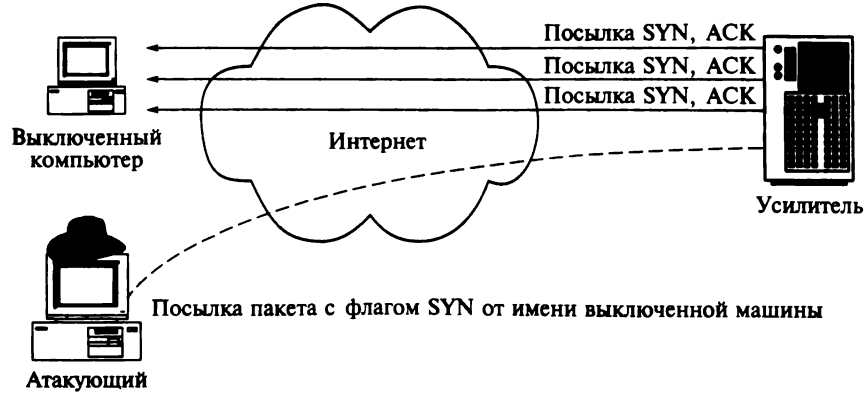


Рис. 3.8. Схема простого усиления атаки

Для защиты от этого вида атак повсеместно на маршрутизаторах устанавливается запрет циркулярной рассылки. Но злоумышленники воспользовались следующей особенностью функционирования ОС. При установлении TCP-соединения (получив пакет с установленным флагом SYN) система отвечает пакетом с установленными флагами SYN и ACK, на который должен последовать пакет с флагом SYN от машины, устанавливающей соединение. Если же последний пакет не приходит, то система повторяет посылку пакета с SYN и ACK, считая, что он не доставлен по техническим причинам. Различные ОС по умолчанию посылают различное число таких пакетов. Например, отдельные версии Linux и Free BSD посылают пять таких пакетов, а Windows — три. Таким образом, злоумышленник получает усиление атаки без использования широковещательных запросов (рис. 3.8).

Подобные атаки получили название *распределенных рефлекторных атак «отказ в обслуживании»* (Distributed Reflection Denial of Service, DRDoS).

3.3.3. Таксономия атак «отказ в обслуживании» и защитных механизмов

С точки зрения организации защиты и разработки методов противодействия распределенным атакам «отказ в обслуживании» целесообразно классифицировать данные атаки. Существует несколько подходов к такой классификации, которые постоянно модифицируются и совершенствуются. Рассмотрим некоторые параметры, по которым можно классифицировать такие атаки.

По используемым уязвимостям атаки можно разбить на два основных класса: атаки на уязвимости (так называемые семантические атаки) и атаки затопления (атаки грубой силы).

Атаки на уязвимости используют один или несколько дефектов в механизмах реализации политики безопасности или ошибки в ПО, применяемом на целевой системе, и реализуются путем посылки специально сформированных пакетов.

Атаки грубой силы пытаются лишить доступа к службе законных пользователей путем посылки громадного количества правильных запросов на предоставление службы.

По распределению источников атаки различают атаки, исходящие из одного источника или из множества источников.

По динамике атаки можно выделить следующие виды атак:

- атаки с постоянной скоростью посылки пакетов;
- атаки с переменной скоростью посылки пакетов (нарастающие, убывающие, периодичные и т. п.);
- атаки, применяющие случайно пульсирующий трафик.

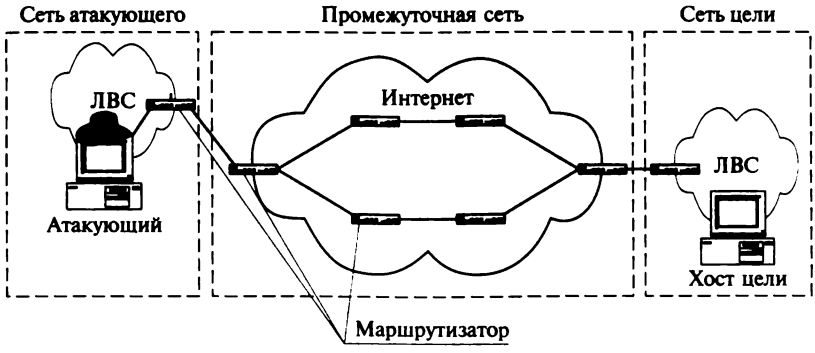


Рис. 3.9. Схема размещения средств защиты:
ЛВС — локальная вычислительная сеть

Рис. 3.10. Методы защиты от распределенных атак «отказ в обслуживании»

Данные параметры не исчерпывают все многообразие распределенных атак «отказ в обслуживании» и могут дополняться производителями средств защиты.

Поскольку защита должна быть эшелонированной, то и защитные механизмы должны располагаться в различных местах защищаемой сети. Специалисты предлагают четыре возможных места расположения средств защиты: в сети защищаемой системы, промежуточной сети, сети источника атаки и во множественных местах. Упрощенная схема сетей, где должны располагаться средства защиты, приведена на рис. 3.9.

Для защиты в сети источника атаки может использоваться так называемая фильтрация ingress (внутренняя) (RFC 2827 «Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing»). Такую фильтрацию может осуществлять маршрутизатор сети или провайдер, которые должны разрешать выходить трафику только в том случае, если адреса источников пакетов соответствуют адресам локальной сети. Аналогичным образом (для исходящего трафика) должна осуществляться фильтрация egress (внешняя).

Общую стратегию защиты от распределенных атак «отказ в обслуживании» можно разделить на следующие три категории: предупреждение, обнаружение и реакция. Таксономия защитных механизмов от распределенных атак «отказ в обслуживании» приведена на рис. 3.10.

3.4. Примеры атак

Для пояснения механизмов действия сетевых атак будем использовать записи журналов программы windump.

Программа tcpdump для исследования дампов сетевого трафика была разработана компанией Network Research Group в лаборатории Lawrence Berkeley National Lab. Она представляет собой набор средств, которые дают возможность исследования дампов сетевого трафика с различными уровнями детализации. На ее основе была создана программа windump, которая выполняет те же функции, но имеет дополнительные опции (http://www.winpcap.org/windump/). В качестве примера рассмотрим элементы записи программы windump в журнал регистрации:

```
15:51:54.781475 IP 194.85.97.54.1076 >
194.85.97.56.21:
%S 2739395289:2739395289(0) win 16384
%<mss 1460,nop,nop,sackOK> (DF)
```

Рассмотрим данную запись поэлементно:

Для *IP 194.85.97.54.1076 > 194.85.97.56.21:*

15:51:54.781475 — значение времени анализа пакета;
IP — контролируемый протокол;
194.85.97.54.1076 — адрес и порт отправителя;
> — направление трафика;
194.85.97.56.21 — адрес и порт получателя;
: — признак окончания адресной части;
% — признак продолжения записи строки журнала.

Для *S 2739395289:2739395289(0) win 16384*

S — значение установленных TCP-флагов (могут быть следующие: P, R, S, F. Здесь S — это SYN);
2739395289:2739395289 — начальный и завершающий порядковые номера;
(0) — количество байтов в пакете;
win 16384 — размер окна передачи.

Для *<mss 1460,nop,nop,sackOK>*

< > — ограничивающие TCP-опцию скобки, параметры опции разделяются запятыми;
mss 1460,nop,nop — максимальный размер сегмента (Maximum Segment Size) для данного соединения. Так как длина этой опции меньше 32 бит, то для выравнивания свободное место заполняется нулями (nop — No Operation code);
sackOK — параметр sackOK;
(DF) — запрет фрагментации.

При получении фрагментированного пакета в записи журнала появляется запись вида *(frag 1109:32@0+)*, где

frag — признак фрагмента пакета;
1109 — идентификатор фрагмента (значение берется из поля идентификатора IP и будет одним и тем же для всех фрагментов данного IP-пакета);
32 — длина содержимого фрагмента в байтах без учета заголовка IP, который занимает 20 байт;
@ — разделитель значений длины фрагмента и смещения;
0 — содержимое фрагмента смещено на 0 байт;
+ — указывает на наличие дополнительных фрагментов для данного IP-пакета (в исходном пакете установлен бит MF).

Для следующего фрагмента того же исходного пакета запись в журнале будет содержать строку вида *(frag 1109:16@32)*, где

1109 — тот же идентификатор, так как фрагменты относятся к одному исходному пакету;
16 — длина фрагмента (без заголовка IP).

Отсутствие знака «+» означает, что это последний фрагмент.

3.4.1. Атаки на протокол IP

Атака *Land* заключается в посылке пакета TCP с установленным SYN на открытый порт. В пакете адрес источника равен адресу назначения, а также указываются одинаковые номера портов:

```
12:00:01:1000 target.com:1056 > target.com:1056
12:00:00:1013 192.168.1.1.80 > 192.168.1.1.80
12:00:00:1025 192.168.1.1.31337 > 192.168.1.1.31337
```

Атака *Christmas Tree* использует слабости программ обработки флагов сетевых пакетов. В данной атаке злоумышленник устанавливает группу флагов, которые не могут использоваться совместно:

```
attack.com:1056 > target.com:1215 UNKNOWN & *1*F*PAU
RESERVEBITS
```

Существует множество атак, основанных на фрагментации. Суть этих атак заключается в использовании ошибок в программной реализации стека TCP/IP в различных ОС. Для этого злоумышленники могут использовать:

- нарушение последовательности прихода фрагментов;
- пропуск в области данных;
- превышение общей длины дейтаграммы;
- незавершенность посылки фрагментов;
- наложение фрагментов;
- дублирование фрагмента без посылки завершающего фрагмента и т.д.

Например, при проведении атаки Teardrop злоумышленник посылает несколько фрагментированных пакетов, в которых последующие фрагменты перекрывают принятые ранее:

```
00:25:48 wild.com.45958 > target.com.3964: udp 28
(frag 242:36@0+)
00:25:48 wild.com > target.com: (frag 242:4@24)
```

В этом примере посылается фрагмент номер 242 с 36 октетами данных со смещением 0. Во второй строке — дополнительные 4 октета данных со смещением 24. Таким образом, чтобы обработать этот пакет, система должна вернуться от 36 к 24.

3.4.2. Атаки на протокол ICMP

Атака сканирования активных портов на целевой системе заключается в посылке запросов всем хостам конкретной подсети:

```
scanner.net > 192.168.1.233: icmp: echo request
scanner.net > 192.168.1.139: icmp: echo request
scanner.net > 192.168.1.242: icmp: echo request
. . .
scanner.net > 192.168.1.63 : icmp: echo request
```

В случае атаки *Smurf* злоумышленник вводит широковещательные эхо-запросы с ложным адресом источника (использует адрес хоста жертвы). Хост жертвы получает большое число эхо-ответов. Эта атака является классической атакой «отказ в обслуживании», ее часто называют ICMP flood. Цель атаки — исчерпать системные ресурсы целевой системы:

```
00:00:05.327 spoofed_1.com > 192.168.15.255: icmp:
echo request
00:00:05.335 spoofed_2.com > 192.168.15.255: icmp:
echo request
00:00:05.368 spoofed_3.com > 192.168.15.255: icmp:
echo request
00:00:05.425 spoofed_4.com > 192.168.15.255: icmp:
echo request
```

Атака *Ping of Death* состоит в посылке большого пакета ICMP с помощью команды *ping*. Команда *ping* использует ICMP для проверки доступности адресата посылкой *ECHO_REQUEST* и ожиданием отклика. Пакеты *ping* имеют длину 32 байт.

```
Для Winows:     ping -l 65527 target.com
Для Unix:       ping -s 65527 target.com
```

Заголовок IP имеет длину 20 байт, поэтому результирующий пакет (65 527 + 32) будет превышать максимально разрешенный размер пакета.

3.4.3. Атаки на протокол UDP

Атака UDP flood (Fraggle) состоит в посылке большого числа пакетов на открытый на целевой системе порт, используя подложные адреса источника. Цель атаки состоит в попытке исчерпания системных ресурсов целевой системы:

```
00:00:01.100 spoofed_1.com > 192.168.2.255: udp
00:00:01.120 spoofed_2.com > 192.168.2.255: udp
00:00:01.131 spoofed_3.com > 192.168.2.255: udp
...
```

Одним из вариантов данной атаки является посылка пакетов UDP на несуществующий порт.

3.4.4 Атаки на протокол TCP

При *атаке сканирования с использованием ACK* флаг ACK, без других установленных флагов, используется только при передаче подтверждений в установленном соединении:

```
00:00:10.256 attack.com:230 > 192.168.1.230: .ack
778483003 win 1028
00:00:10.322 attack.com:143 > 192.168.1.143: .ack
778483003 win 1028
```

Атака *TCP flood* (SYN flood) является типичной атакой «отказ в обслуживании». В этой атаке злоумышленник посылает множество пакетов для установления TCP-соединения. При получении каждого такого пакета целевая система выделяет буфер для приема последующих данных. Количество таких буферов определяется каждой ОС, и это количество ограничено. В результате атаки целевая система оказывается не способной функционировать:

```
attack.com:1076 > target.com:21: S
2739395289:2739395289(0)
win 16384 <mss 1460, nop,nop,sackOK>
. . .
attack.com:1076 > target.com:21: S
2739395289:2739395289(0)
win 16384 <mss 1460, nop,nop,sackOK>
```

Атака *WinNuke* также является атакой «отказ в обслуживании». Обычно атака заключается в посылке пакета TCP на 139 порт и установке флага Urgent. Когда система Windows встречает флаг Urg, она готовится к приему данных. Вместо этого злоумышленник отправляет пакет с флагом RST (Reset) для разрыва соединения:

```
attack.com:1457 > target.com:139: S 49154:49154(0)
win 8192 <mss 1460> (DF)
target.com:139 > attack.com:1457: S 42414:42414(0)
ack 49155 win 8760 <mss 1460> (DF)
attack.com:1457 > target.com:139: .ack 1 win 8760
(DF)
attack.com:1457 > target.com:139: P 1:4(3) ack 1
win 8760 urg 3 (DF)
target.com:139 > attack.com:1457: FP 1:6(5) ack 4
win 8758 (DF)
attack.com:1457 > target.com:139: .ack 7 win 8755 (DF)
```

```
attack.com:1457 > target.com:139: R 49158:49158(0)
win 0 (DF)
```

Множество подобных атак базируется на использовании комбинаций флагов, не описанных в RFC. Таким образом, атакующий проверяет наличие уязвимостей в программной реализации стека TCP/IP. Приведем список (неполный) нормальных появлений флагов TCP и их комбинаций:

SYN; SYN,ACK; ACK (при установлении соединения); *RST; RST,ACK; FIN; FIN,ACK; FIN,PSH,ACK; URG,ACK; PSH,ACK.*

3.4.5. Генераторы атак

Использование одного вида атак во многих случаях оказалось неуспешным, так как были разработаны соответствующие меры противодействия. Поэтому злоумышленники формировали средства, способные осуществлять множество атак из одного источника. Такие средства получили название *генераторов атак*. В качестве примера приведем заголовок программной реализации генератора targa 2:

```
/* targa2.c - copyright by Mixter <mixter@popmail.
com>
   version 2.1 - released 22/3/99 - interface to 11
   multi-platform remote denial of service exploits
   *
   * featured exploits / authors / vulnerable plat-
forms
   * bonk by route|daemon9 & klepto          -     win95,
nameservers
   * jolt by Jeff W. Roberson (Mixter)       -     win95,
klog (old linux)
   * land by m3lt - win95/nt, old un*x's
   * nestea by humble & ttol - older linux/bsd?
   * newtear by route|daemon9 - linux/bsd/win95/others
   * syndrop by PineKoan - linux/win95/?
   * teardrop by route|daemon9 - lots of os's
   * winnuke by _eci - win95/win31
   * 1234 by DarkShadow/Flu - win95/98/nt/others?
   * saihyousen by noc-wage - win98/firewalls/routers
   * oshare by r00t zer0 - win9x/NT/macintosh
   */
```

Как видно из кратких комментариев, это средство может реализовать 11 атак, каждая из которых является разновидностью атак DoS.

3.4.6. Атака К. Митника

Часто атаки различных типов используются совместно для достижения атакующим своей цели. В качестве примера такой комбинированной атаки рассмотрим атаку К. Митника (Kevin Mitnick), схема которой показана на рис. 3.11. Проиллюстрируем основные этапы этой атаки данными tcpdump, которые приведены Ц. Шимомурой (Tsutomu Shimomura).

В приведенных данных используются следующие обозначения: *server* — рабочая станция SPARCstation; *terminal* (*x-terminal*) — бездисковая рабочая станция. Для краткости часть записей опущена (указывается многоточием).

В ходе атаки К. Митник выполнил следующие основные действия:

1. Проведение проб (с адреса toad.com) атакуемых машин для определения наличия доверительных отношений между ними:

```
14:10:21 toad.com# finger -l @server
14:10:50 toad.com# finger -l root@server
14:11:07 toad.com# finger -l @x-terminal
14:11:38 toad.com# showmount -e x-terminal
14:11:49 toad.com# rpcinfo -p x-terminal
14:12:05 toad.com# finger -l root@x-terminal
```

2. Инициирование атаки отказа в обслуживании (SYN flood) против сервера, чтобы воспрепятствовать ответам сервера на запросы терминала. Адрес источника 130.92.6.97 выбран из неиспользуемых или неактивных в тот момент адресов, чтобы данный хост не отвечал на получаемые пакеты (атака проводилась в праздничные дни 1994 г.):

```
  14:18:22.516699 130.92.6.97.600 > server.login: S
1382726960:1382726960(0) win 4096
  14:18:22.566069 130.92.6.97.601 > server.login: S
1382726961:1382726961(0) win 4096
  14:18:22.744477 130.92.6.97.602 > server.login: S
1382726962:1382726962(0) win 4096
  Пропущены записи для краткости.
  14:18:25.483127 130.92.6.97.627 > server.login: S
1382726987:1382726987(0) win 4096
  14:18:25.599582 130.92.6.97.628 > server.login: S
1382726988:1382726988(0) win 4096
  14:18:25.653131 130.92.6.97.629 > server.login: S
1382726989:1382726989(0) win 4096
```

Сервер сгенерировал ответы (SYN и ACK) на первые восемь запросов, после чего очередь запросов была заполнена.

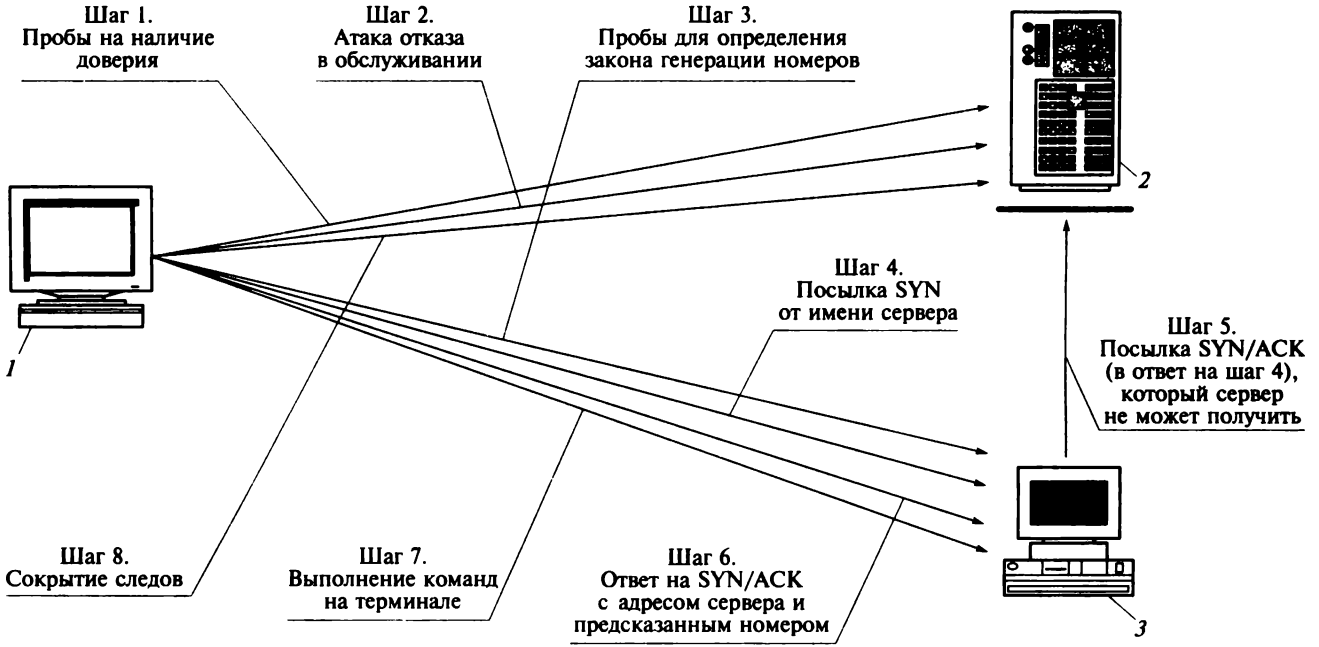


Рис. 3.11. Схема атаки Митника:
1 — атакующий; *2* — сервер; *3* — терминал

3. Посылка 20 запросов (от apollo.it.luc.edu) на установление соединения с терминалом для определения поведения генератора последовательных номеров (sequence number) TCP-соединений терминала:

```
14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.
shell: S 1382726990:1382726990(0) win 4096
14:18:26.094731 x-terminal.shell > apollo.it.luc.
edu.1000: S 2021824000:2021824000(0) ack 1382726991
win 4096
14:18:26.172394 apollo.it.luc.edu.1000 > x-terminal.
shell: R 1382726991:1382726991(0) win 0
14:18:26.507560 apollo.it.luc.edu.999 > x-terminal.
shell: S 1382726991:1382726991(0) win 4096
14:18:26.694691 x-terminal.shell > apollo.it.luc.edu.
999: S 2021952000:2021952000(0) ack 1382726992 win 4096
14:18:26.775037 apollo.it.luc.edu.999 > x-terminal.
shell: R 1382726992:1382726992(0) win 0
...
14:18:35.225869 apollo.it.luc.edu.982 > x-terminal.
shell: S 1382727008:1382727008(0) win 4096
14:18:35.395723 x-terminal.shell > apollo.it.luc.edu.
982: S 2024128000:2024128000(0) ack 1382727009 win
4096
14:18:35.472150 apollo.it.luc.edu.982 > x-terminal.
shell: R 1382727009:1382727009(0) win 0
14:18:35.735077 apollo.it.luc.edu.981 > x-terminal.
shell: S 1382727009:1382727009(0) win 4096
14:18:35.905684 x-terminal.shell > apollo.it.luc.edu.
981: S 2024256000:2024256000(0) ack 1382727010 win
4096
14:18:35.983078 apollo.it.luc.edu.981 > x-terminal.
shell: R 1382727010:1382727010(0) win 0
```

Из приведенных записей видно, что каждый пакет с SYN и ACK, посланный терминалом, имеет начальный последовательный номер на 128 000 больше предыдущего (значения выделены жирным шрифтом), если это соединение будет следующим. Установление данного факта позволяет атакующему предсказать очередной номер, не получая самого пакета.

4. Инициирование соединения с терминалом от имени сервера в предположении, что терминал доверяет серверу:

```
14:18:36.245045 server.login > x-terminal.shell: S
1382727010:1382727010(0) win 4096
```

5. При получении запроса терминал посылает серверу ответ (SYN и ACK), который должен быть подтвержден сервером (ACK) для установления соединения. Так как сервер не посылал пакета на установление соединения, то должен был бы ответить RST-пакетом и разорвать установку соединения. Но в результате начавшейся 14 с раньше атаки сервер наводнен запросами на установку соединений и не может ответить.

6. Атакующий посылает пакет с ACK от имени сервера с предсказанным номером (без получения ответа с SYN и ACK):

```
14:18:36.755522 server.login > x-terminal.shell: .
ack 2024384001 win 4096
```

7. Пока сервер заблокирован, соединение с доверительными отношениями используется для выполнения следующей команды:

```
14:18:37 server# rsh x-terminal «echo + + >>/.
rhosts»
```

Эта команда указывает терминалу доверять всем хостам и всем пользователям этих хостов:

Машина атакующего, таким образом имеет соединение с терминалом, которое установлено от имени сервера. Атакующий может сохранять соединение, посылая необходимые подтверждения терминалу. Атакующий посылает следующие пакеты:

```
14:18:37.265404 server.login > x-terminal.shell: P
0:2(2) ack 1 win 4096
14:18:37.775872 server.login > x-terminal.shell: P
2:7(5) ack 1 win 4096
14:18:38.287404 server.login > x-terminal.shell: P
7:32(25) ack 1 win 4096
```

Эти пакеты соответствуют выполнению на терминале следующей команды:

```
14:18:37 server# rsh x-terminal «echo + + >>/.
rhosts»
```

Теперь К.Митник имеет возможность подключиться к терминалу с любого хоста и выполнить на терминале любую команду.

8. Закрываются все соединения с терминалом от имени сервера:

```
14:18:41.347003 server.login > x-terminal.shell: .
ack 2 win 4096
14:18:42.255978 server.login > x-terminal.shell: .
ack 3 win 4096
```

```
14:18:43.165874 server.loginserver.login > x-terminal.
shell: F 32:32(0) ack 3 win 4096
  14:18:52.179922 server.login > x-terminal.shell: R
1382727043:1382727043(0) win 4096
  14:18:52.236452 server.login > x-terminal.shell: R
1382727044:1382727044(0) win 4096
```

Закрываются все полуоткрытые запросы на установление соединения с сервером, чтобы другие пользователи не обнаружили безуспешность попыток установления соединений:

```
  14:18:52.298431 130.92.6.97.600 > server.login: R
1382726960:1382726960(0) win 4096
  14:18:52.363877 130.92.6.97.601 > server.login: R
1382726961:1382726961(0) win 4096
  14:18:52.416916 130.92.6.97.602 > server.login: R
1382726962:1382726962(0) win 4096
  14:18:52.476873 130.92.6.97.603 > server.login: R
1382726963:1382726963(0) win 4096
```

Сервер теперь функционирует в обычном режиме, т.е. готов отвечать на запросы установления соединений.

Таким образом, для проведения атаки К.Митник использовал недостатки протокола TCP, которые были хорошо известны по публикациям, но не учитывались разработчиками систем.

3.5. Классификации удаленных атак

Для того чтобы защититься от атак, необходимо их изучать и классифицировать. Систематизация знаний об атаках помогает разработке мер и систем защиты от них. Поэтому специалисты в области информационной безопасности не прекращают попыток построения различных классификационных схем, которые в той или иной мере способствуют пониманию процессов, ведущих к проникновению в системы, и помогают разрабатывать меры защиты и реализовывать системы защиты. В качестве примеров построения таких классификационных схем рассмотрим списки терминов, списки категорий, матричные схемы, процессы Столингса, таксономические схемы Ховарда и онтологию сетевых атак.

3.5.1. Списки терминов

В качестве примера приведем часть списка наиболее часто употребляемых терминов, предложенного Коэном:

backup theft	— кража резервных копий
combined attacks	— комбинированные атаки
computer viruses	— компьютерные вирусы
data aggregation	— агрегирование данных
e-mail overflow	— переполнение почты
e-mail spoofing	— обман почты
fictitious people	— фиктивные люди
human engineering	— социальная инженерия
illegal value insertion	— вставка недопустимых значений
infrastructure interference	— помехи инфраструктуре
infrastructure observation	— наблюдение инфраструктуры
input overflow	— переполнение буфера при вводе
login spoofing	— обман регистрации
network services attacks	— атаки на сетевые службы
packet insertion	— вставка пакета
packet watching	— просмотр пакета
password guessing	— угадывание (подбор) пароля
process bypassing	— обход процесса
protection limit poking	— нарушение пределов защиты
shoulder surfing	— подсматривание через плечо
time bombs	— временные бомбы
trojan horses	— троянские кони

Необходимо отметить, что данный подход не является таксономией, так как отдельные термины перекрываются, а отдельные атаки не включены в список.

3.5.2. Списки категорий

Списки категорий являются вариацией списка терминов, но содержат определения категорий. Примером может послужить список категорий, данный Чезвиком и Беллоуином (Cheswick and Bellovin). Они подразделяли атаки на следующие семь категорий:

1) кража паролей (stealing passwords) — методы получения паролей пользователей;

2) социальная инженерия (social engineering) — использование некоторых психологических приемов по отношению к пользователям для получения желаемой информации или информации ограниченного использования;

3) ошибки и потайные ходы (bugs and backdoors) — поиск состояния системы, не соответствующего спецификации, или перезапись компонентов ПО скомпроментированными компонентами;

4) отказы аутентификации (authentication failures) — удаление механизмов, использующихся для аутентификации;

5) отказы протоколов (protocol failures) — протоколы сами по себе либо плохо спроектированы, либо плохо реализованы;

6) утечка информации (information leakage) — использование таких систем, как finger или DNS, для получения информации, необходимой администраторам для надлежащего функционирования сети, но используемой атакующим;

7) отказ в обслуживании (denial-of-service) — усилия для прекращения возможности пользователей пользоваться службами.

В данном подходе также существует перекрытие понятий, что потребовало применения списков внутри категорий.

3.5.3. Матричные схемы

Матричные схемы базируются на нескольких измерениях. Для двух измерений рассматриваются уязвимости и потенциальные нарушители. Матричный подход был реализован Ландвейром (Landwehr). Он представил таксономию уязвимостей (условий, которые могут привести к отказу в обслуживании или неавторизованному доступу), базирующуюся на трех измерениях:

- происхождение (genesis) — как уязвимости находят свою дорогу к программам;
- время внедрения (time of introduction) — в жизненном цикле ПО или аппаратуры;
- местоположение (location) — расположению в ПО или аппаратуре.

Этот подход иллюстрируется табл. 3.4.

Данная таксономия нашла свое применение при разработке систем обнаружения вторжений.

3.5.4. Процессы

Столлингс (Stallings) разработал простейшую модель, представляющую классификацию угроз безопасности. Модель опирается на передачу информации между объектами. Столлингс определил четыре категории атак (рис. 3.12).

1. Прерывание потока — объект системы разрушен или стал недоступен.

2. Перехват потока — неавторизованный субъект (объект) получил доступ к объекту.

3. Модификация потока — неавторизованный субъект (объект) не только получил доступ к объекту, но и изменил его.

4. Подделка потока — неавторизованный субъект (объект) вставил поддельный объект в систему.

Эту наглядную классификацию можно дополнить категорией атаки отказа в обслуживании (например, заполнение потока).

Таблица 3.4. **Матричная классификация Ландвейра**

Происхождение (Genesis)	Злоумышленные (Intentional)	Зловредные (Malicious)	Троянские кони (Trojan Horse)	Некопирующиеся (Non-Replicating)
				Копирующиеся (Replicating)
			Потайные ходы (Trapdoor)	
			Логические/Временные бомбы (Logic/Time Bomb)	
		Незловредные (Non-Malicious)	Скрытые каналы (Covert Channel)	Storage
				Временные (Timing)
			Другие (Other)	
	Случайные (Inadvertent)	Ошибки проверок (Неполнота/Несовместимость) Validation Error (Incomplete/Inconsistent)		
		Ошибки домена (включая повторное использование объекта, остатки и ошибки открытого представления) Domain Error (Including Object Re-use, Residuals, and Exposed Representation Errors)		
		Сериализации/Замены (Serialization/Aliasing)		
		Неадекватная идентификация/аутентификация (Identification/Authentication Inadequate)		
		Нарушение границ (включая исчерпывание ресурса и ошибки нарушения ограничений) Boundary Condition Violation (Including Resource Exhaustion and Violable Constraint Errors))		
		Другие ошибки логики применения (Other Exploitable Logic Error)		

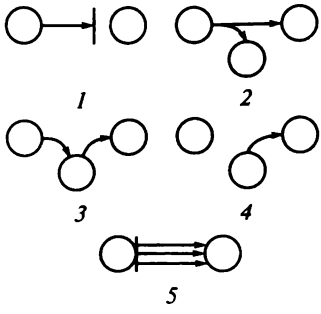


Рис. 3.12. Модель Столингса:
1 — прерывание потока; *2* — перехват потока; *3* — модификация потока; *4* — подделка потока; *5* — заполнение потока

3.5.5. Классификация Ховарда

Таксономия атак разработана Ховардом (Howard) на основе анализа статистики инцидентов CERT с 1989 по 1995 г. Таксономическая схема Ховарда представлена на рис. 3.13.

Согласно таксономии Ховарда *угрозой* являются:

- хакеры — лица, вторгающиеся в компьютерные системы для получения доступа с целью вызова и утверждения своего статуса;
- шпионы — лица, вторгающиеся в компьютерные системы для получения информации, которая может быть использована в политических целях;
- террористы — лица, вторгающиеся в компьютерные системы для того, чтобы вызвать страх, который поможет им достигнуть политических целей;
- конкуренты — лица (пользователи одной компании), вторгающиеся в чужие компьютерные системы для получения финансовой выгоды для организации;
- криминал — профессиональные преступники, вторгающиеся в компьютерные системы для получения личной финансовой выгоды;
- вандалы — лица, вторгающиеся в компьютерные системы для причинения ущерба.

Средства проведения атак:

- пользовательские команды — атакующий вводит команды в командной строке или задает их с помощью графического интерфейса пользователя;
- скрипт или программа — атакующий разрабатывает (составляет) скрипты и (или) программы, инициируемые с помощью интерфейса пользователя для использования уязвимости;
- автономные агенты — атакующий инициализирует программу или фрагмент программы, которая функционирует независимо от пользователя, используя уязвимости;
- набор средств — атакующий использует пакет программ, который содержит скрипты, программы или автономные агенты для использования уязвимостей;

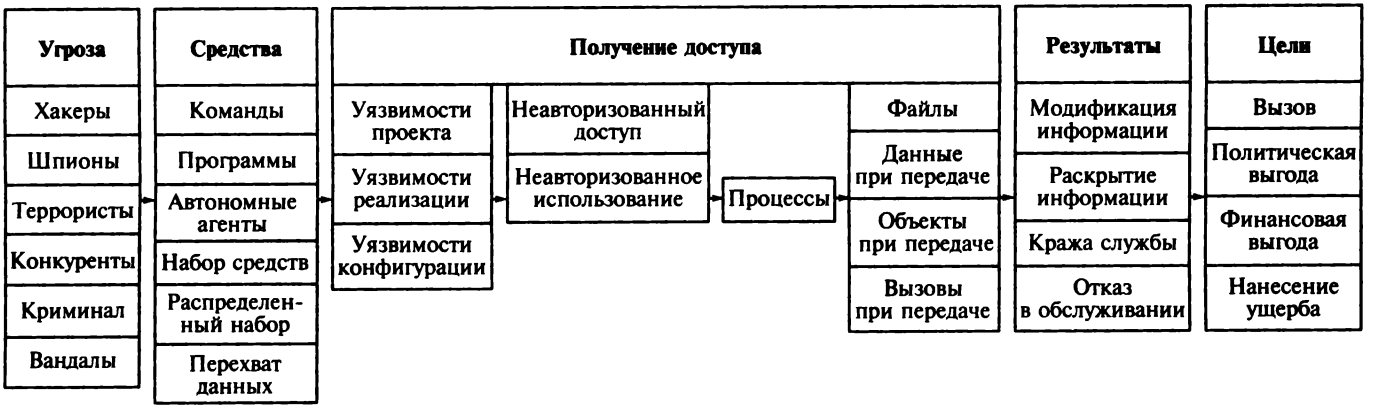


Рис. 3.13. Таксономическая схема Ховарда

• распределенный набор средств — атакующий распределяет средства по множеству хостов, которые после некоторой задержки скоординированно атакуют целевой хост одновременно;

• перехват данных — в этом случае атакующий либо перехватывает технические каналы утечки (например, через электромагнитное излучение), либо перехватывает («слушает») трафик.

Получение доступа реализуется за счет использования:

• уязвимостей проекта, реализации или применения (конфигурирования);

• неавторизованного доступа или неавторизованного использования;

• процессов, работающих с файлами, данными при передаче, объектами или вызовами при передаче.

Результатами атак являются:

• модификация информации — любое неавторизованное изменение файлов, сохраняемых в компьютере, или изменение данных, передаваемых по сети;

• раскрытие информации — рассылка информации кому-либо, кто не авторизован для доступа к ней;

• кража службы — неавторизованное использование компьютера или сетевой службы без деградации службы для других пользователей;

• отказ в обслуживании — умышленная деградация или блокировка компьютера или сетевого ресурса.

Целями проведения атак являются:

• вызов, статус;

• политическая выгода;

• финансовая выгода;

• ущерб.

К достоинствам данной классификации можно отнести достаточно хорошую проработку категорий. Недостатки таксономии определяются ее целью — построением классификации для уже осуществленных атак. Позднее Ховард совместно с Лонгстафом (Longstaff) разработал уточненную таксономию, представленную на рис. 3.14.

Данная таксономия была разработана в 1998 г. для языка описания компьютерных инцидентов и является расширением предыдущей таксономии.

Представленные таксономии атак предназначены для описания произошедших атак и могут быть полезны при разработке систем обнаружения вторжений. В этой таксономии задачи и действия атакующего составляют событие, которое может быть обнаружено в атакуемой системе. Атака, согласно данной таксономии, помимо события включает в себя используемое средство, используемую уязвимость и полученный результат. Все элементы таксономии представляют собой инцидент.

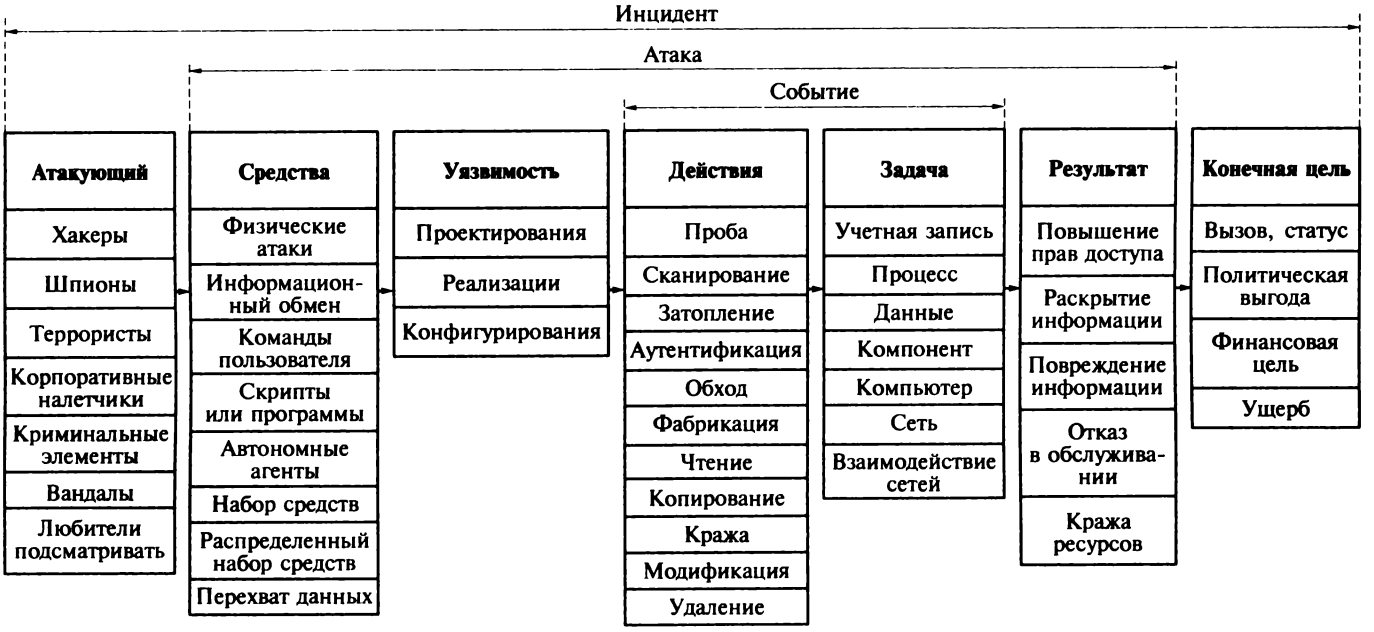


Рис. 3.14. Таксономия Ховарда и Лонгстафа

Дальнейшим шагом в построении таксономий атак явились попытки построения онтологий атак.

3.5.6. Построение онтологии сетевых атак

Онтология (от древнегреческого онтос — сущее, логос — учение, понятие) — философский термин, определяющий учение о бытии. Это учение восходит к произведениям Аристотеля, Фомы Аквинского, Х. Вольфа, а в XX в. его развивал М. Хайдеггер. Он считал, что онтология возможна не иначе как герменевтика, т. е. как наука об интерпретации, осуществляемой в мысли и языке. Это положение прослеживается и в спецификациях Международной федерации по разработке интеллектуальных физических агентов (FIPA98 Specifications. Part 12. Ontology Service: http://www.cset.it/fipa/), где под онтологией понимается явное описание структуры некоторой проблемной области (темы). Подобное описание всегда опирается на определенную концепцию этой области, которая обычно задается в виде системы исходных объектов (понятий), отношений между ними и положений (аксиом). Само определение базовых понятий предметной области (агентов, процессов, атрибутов) вместе с основными отношениями между ними называется *концептуализацией*. Поэтому онтологию часто понимают как «спецификацию концептуализации» и даже считают синонимом «концептуальной модели предметной области» (точнее, набора сосуществующих концептуальных моделей). Онтологии представляют собой договоренности — соглашения о совместно используемых концептуализациях.

С одной стороны, в онтологических исследованиях изучаются вопросы происхождения знаний в конкретной предметной области и их конструирования из некоторых единиц. С другой стороны, эти исследования направлены на поддержку процессов коммуникации, предполагающих разделение знаний между агентами и их повторное использование.

В простейшем случае онтология определяется как некоторый общий словарь понятий, используемых в качестве строительных кирпичиков в системах обработки информации. Обычно онтология описывает иерархию понятий, связанных между собой отношениями категоризации. В частности, при взаимодействии агентов в сети Интернет онтология понимается как иерархия понятий и связей между ними вместе с системой ссылок на www-документы, привязанных к этим понятиям (связям).

Модель онтологии должна обеспечивать представление множества понятий в виде сетевой структуры, отображение достаточно богатого множества отношений, включающего в себя не только таксономические отношения, но и отношения, отражающие специфику предметной области, использование декларативных и процедурных

интерпретаций и отношений. Под обобщенной формальной моделью онтологии понимается тройка

$$ONT = \{U, Im(R), F\},$$

где U — множество понятий предметной области, $|U| \neq \varnothing$; $Im(R)$ — множество нечетких (взвешенных) отношений между понятиями предметной области, $Im(R) = \{W | w: U^n \rightarrow [0,1]\}$; F — конечное множество функций интерпретации (аксиоматизаций), заданных на понятиях и(или) отношениях онтологии, $F = \{f\}, f: D^n \rightarrow [0,1]$, D — область интерпретации.

Рассмотрим применение онтологии для построения классификации атак с точки зрения цели атаки. Высокоуровневое представление атак включает в себя следующие свойства: цель атаки, средство атаки, результат атаки и расположение источника атаки.

Согласно данному высокоуровневому представлению вторжение представляет собой некоторый ввод данных, который получен из некоторого местоположения, направлен на определенный системный компонент, использует некоторый метод и вызывает некоторое системное поведение (рис. 3.15).

Полная онтология атак в графической форме представлена на рис. 3.16.

Рассмотрим характеристики и свойства данной онтологии.

1. Системный компонент, который является целью атаки:

- сетевой протокол (атака использует протоколы стека протоколов и не выходит за них);
- пространство ядра (процесс, выполняющийся как часть ОС, который или компилируется в ядро, или загружается модулем и используется в ядре);
- приложение (приложение, выполняющееся вне пространства ядра с привилегиями пользователя или root);
- другие (не указанные выше, например, принтеры, модемы).

2. Метод, используемый атакующим:

- ошибка проверки ввода, которая включает в себя: переполнение буфера, нарушение границ (процесс может читать или писать за раз-

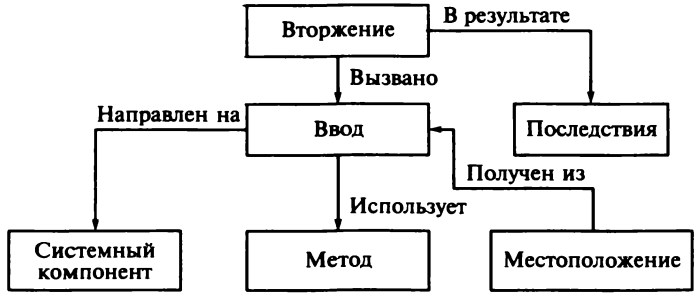


Рис. 3.15. Высокоуровневое представление атаки

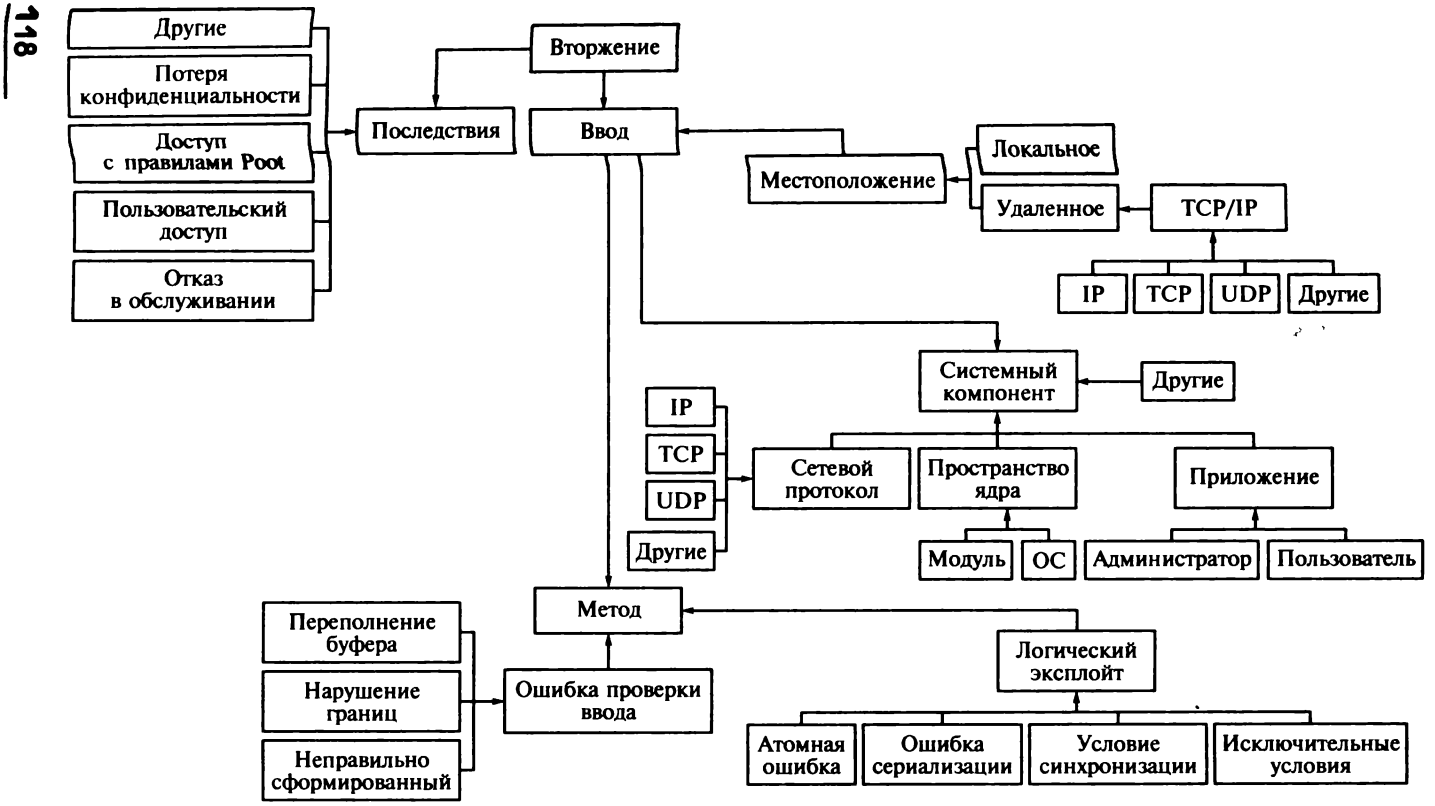


Рис. 3.16. Графическое представление онтологии атак

решенное пространство адресов либо исчерпывает системный ресурс), неправильно сформированный ввод (процесс допускает синтаксически некорректный ввод, ненормальные поля или не имеет возможности корректировать ошибки);

• логический эксплойт, использующий уязвимость и ведущий к снижению производительности и (или) компрометации системы: исключительные условия (ошибки, вызванные отказом обработки исключительных условий, сгенерированных функциональным модулем или устройством), условие синхронизации (ошибки, возникающие во время временно́го промежутка между двумя операциями), ошибки сериализации в результате неправильных операций сериализации, атомные ошибки (ошибки, возникающие, когда частично модифицированные одним процессом данные уже используются другим процессом).

3. Последствия — конечный результат атаки:

• отказ в обслуживании пользователей системы;

• пользовательский доступ (атакующий получает доступ к некоторым службам целевой системы);

• доступ с правами root (атакующий получает полное управление системой);

• потеря конфиденциальности (в результате атаки пользователь системы теряет конфиденциальность данных);

• другие (результат заключается в компрометации целостности или других нежелательных характеристиках).

4. Местоположение источника атаки — атакующий соединяется через сеть или находится на хосте:

• удаленное (атакующему нет необходимости «виртуально» находиться на цели);

• локальное (атакующему необходимо «виртуально» присутствовать на цели);

• удаленно-локально (атакующий на разных стадиях атаки может быть как удаленно, так и локально).

3.6. Оценивание степени серьезности атак

Для оценки серьезности атак обычно принимаются простые показатели. Главной сложностью при определении показателя является присвоение значений каждой атаке. Как правило, такое назначение осуществляется экспертами, которые имеют достаточный опыт работы с соответствующими показателями. В качестве примера такого показателя рассмотрим параметр важности атаки, который используется в CERT и его обучающем центре.

Данный показатель состоит из трех параметров:

• простота (значения от 1 до 10; 10 — самый простой);

• сложность (от 1 до 10; 1 — самый сложный);
• доступность (от 1 до 10; 10 — самый доступный).

Значение каждого показателя определяется как среднее арифметическое.

Всемирный центр анализа происшествий (Global Incident Analysis Center, GIAC) уже упоминавшегося института SANS для оценки атак использует показатель серьезности атаки. Общее представление об уровне серьезности атаки иллюстрируется следующей цепочкой (по нарастанию риска): ненаправленный неэффективный сценарный метод нападения (риска нет) → разведывательное зондирование → направленный метод нападения → поражение не основной системы →поражение основной системы (максимальный риск).

Оценка показателя серьезности атаки *P* определятся следующими параметрами:

• важность цели;
• катастрофичность атаки;
• контрмеры (системные и сетевые).

Для каждого параметра установлены оценочные шкалы, некоторые из них приведены в табл. 3.5.

Значение показателя серьезности атаки вычисляется по формуле

$$P = (V + H) - (C1 + C2).$$

Значение каждого элемента выбирается по 5-балльной шкале, где 1 — минимальное значение, а 5 — максимальное. Максимальное значение степени тяжести последствий (наихудший вариант) равно 8, а минимальное (наилучший вариант) — равно –8. Отрицательные значения свидетельствуют о высокой надежности мер обеспечения безопасности.

Рассмотрим два примера вычисления параметра серьезности атаки.

1. Взлом Boink — приводит к отказу в обслуживании уязвимых хостов за счет использования фрагментации для исчерпания системных ресурсов.

```
07:26:40.754197 25.25.25.25.20 > protect-5:20: udp 28
%(frag 1109:36@0+)
07:26:40.754281 25.25.25.25 > protect-5:
%(frag 1109:4@32)
```

Механизм нарушения. На различные порты направляются два UDP-пакета, содержащих перекрывающиеся фрагменты. Смещение фрагмента во втором пакете (32) меньше, чем размер полезных данных в первом пакете (36). Кроме того, размер данных первого пакета (36) не соответствует правилу фрагментации, согласно которому размер полезных данных всех фрагментов (кроме последнего) должен быть кратен 8.

Значения параметров:

• важность цели — 3 (нападению подвергается сервер системы);

Таблица 3.5. Оценочные шкалы параметров

Показатель	Значение шкалы	Что включает
Важность цели (V)	5	МЭ, сервер DNS, основной маршрутизатор
	4	Ретранслятор (средство) обмена для электронной почты
	3	Пользовательская система Windows
	2	Пользовательская система Unix
	1	MS DOS
Катастрофичность атаки (H)	5	Нарушитель может получить административный доступ управления сетью
	4	Полная блокировка атакой отказа в обслуживании
	3	Пользовательский доступ (например, через пароль)
	1	Вероятность успешной атаки мала
Системные контрмеры ($C1$)	5	ОС с патчами и дополнительной защитой (типа TCP Wrappers, secure shell)
	4	Отсутствие некоторых патчей
	3	Нет TCP Wrappers, разрешены фиксированные незашифрованные пароли
	1	Старые ОС
Сетевые контрмеры ($C2$)	5	Ограничительный МЭ
	4	Ограничительный МЭ с внешними соединениями (модемы, ISDN)
	2	Разрешительный МЭ
	1	Отсутствие МЭ

- катастрофичность атаки — 4 (полная блокировка сервера);
- контрмеры системы — 1 (используется старая ОС);
- контрмеры сети — 1 (нет межсетевого экрана).

Степень тяжести последствий нарушения равна 5.

2. Взлом Teardrop — также приводит к отказу в обслуживании уязвимых хостов.

```
10:13:32.104203 25.25.25.25.53 > 192.168.1.3.53:
%udp 28 (frag 242:36@0+) (ttl 64)
10:13:32.104272 25.25.25.25 > 192.168.1.3:
%(frag 242:4@24) (ttl 64)
```

Механизм нарушения. На целевой хост направляются фрагментированные IP-дейтаграммы, причем второй фрагмент полностью помещается в первом.

Значения параметров:

- важность цели — 2 (нападению подверглась рабочая станция Unix);
- катастрофичность атаки — 1 (система защищена от взломов Teardrop);
- контрмеры системы — 5 (ОС защищена от этого взлома);
- контрмеры сети — 1 (нет межсетевого экрана).

Степень тяжести последствий нарушения равна –3.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите основные угрозы при сетевом взаимодействии.
2. Что понимается под удаленной сетевой атакой?
3. Перечислите основные типы сетевых атак и их особенности.
4. Что может выступать в качестве атакуемого программного средства?
5. Что понимается под уязвимостью и эксплойтом?
6. Какая взаимосвязь между атакой и сценарием атаки?
7. Каково главное отличие атаки от вторжения?
8. Перечислите основные шаги обобщенного сценария атаки.
9. Перечислите обстоятельства, обусловившие высокую скорость распространения червя Slammer.
10. Что может быть установлено на атакуемой системе в результате успешной атаки?
11. Каковы основное назначение и виды rootkits?
12. На чем основаны атаки, использующие фрагментирование пакетов?
13. Перечислите типы отдельных атак, использованных в атаке Митника.
14. Используя приведенные в главе примеры атак, дополните категории атак, введенные Столингсом.
15. Перечислите основные средства атак по классификации Ховарда.
16. Перечислите основные различия онтологии и таксономии.
17. Каковы возможные границы диапазона значений уровня серьезности атак по схеме оценки GIAC?

Глава 4
ТЕХНОЛОГИИ МЕЖСЕТЕВЫХ ЭКРАНОВ

> ...если поглубже полоснуть по пальцу ножом, из пальца обычно идет кровь; если разом осушить пузырек с пометкой «Яд!», рано или поздно почти наверняка почувствуешь недомогание. Последнее правило Алиса помнила твердо.
>
> *Л. Кэрролл. Алиса в стране чудес*

Одним из основных элементов эшелонированной обороны корпоративной сети являются межсетевые экраны. Кроме того, межсетевые экраны являются первым защитным устройством, разделяющим внешний и внутренний периметры.

Межсетевой экран представляет собой локальное (однокомпонентное) или функционально распределенное средство (комплекс), реализующее контроль за информацией, поступающей в автоматизированную систему (АС) и (или) выходящей из нее, и обеспечивает защиту АС посредством фильтрации информации, т. е. ее анализа по совокупности критериев и принятия решения о ее распространении. МЭ просматривает пакеты, проходящие через него в обоих направлениях, и принимает решение о допуске или уничтожении пакетов. Таким образом, МЭ реализует одну точку защиты между двумя сетями — он защищает одну сеть от другой.

4.1. Развитие технологий межсетевого экранирования

Технологии МЭ сравнительно молоды, но быстро развиваются. Приблизительные временны́е рамки развития этих технологий приведены на рис. 4.1.

Первое появление МЭ как технологии связывается с маршрутизаторами, которые появились в 1983—1985 гг. Эти МЭ назывались фильтрами пакетов. Первая статья, описывавшая процесс фильтрации пакетов для целей защиты, опубликована Digital Equipment Corporation в 1988 г.

В 1989—1990 гг. в AT&T Bell Lab появилось второе поколение архитектур МЭ, связанное с исследованием задержек в цепях. При этом исследователи предложили первую рабочую модель третьего поколения — МЭ прикладного уровня, но данные идеи не были детально проработаны и реализованы.

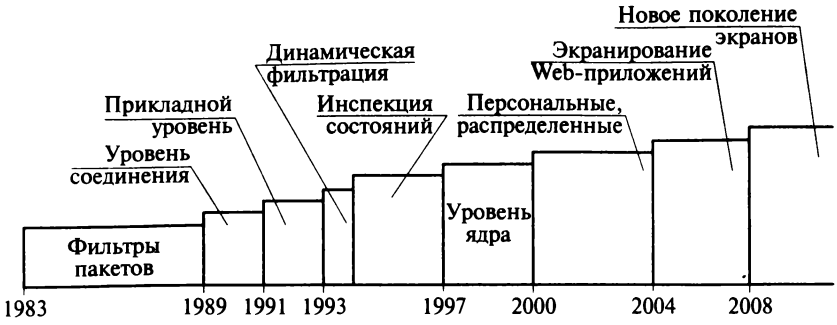


Рис. 4.1. Развитие технологий МЭ

Поэтому межсетевые экраны третьего поколения одновременно предложены несколькими исследователями в конце 1980-х — начале 1990-х гг. В первых публикациях Спаффорд (Gene Spafford) из университета Purdue, Чезвик (Bill Cheswick) из AT&T Bell Lab и Ранум (Marcus Ranum) описали МЭ прикладного уровня в 1990—1991 гг.

В 1991 г. Ранум предложил форму хоста-бастиона, выполняющего службу proxy. Она была вскоре реализована фирмой DEC (продукт SEAL).

В 1991 г. Чезвик и Белловин (Steve Bellovin) начали исследовать динамические фильтры пакетов и разрабатывать в Bell Lab соответствующую архитектуру, но она не была до конца реализована. В 1992 г. Брайден (Bob Braden) и Дешан (Annette Deschan) из USC Informatics Science Institute разработали систему динамической фильтрации «Visas» Check Point Software, реализованную в 1994 г. Данная технология была запатентована компанией под названием «инспекция состояний» (statefull inspection).

В 1996 г. начались работы по разработке 5-го поколения: Kernel Proxy. В 1997 г. был реализован Cisco Centry FW — первый коммерческий МЭ 5-го поколения.

В 1999 г. была предложена идея построения распределенных межсетевых экранов.

Необходимо отметить, что как исследовательские (академические), так и коммерческие МЭ представляли и представляют собой громадные и сложные программные продукты. Стоимость их практически недоступна для рядовых пользователей. Поэтому начиная с 2000 г. широкое направление исследований было нацелено на разработку персональных МЭ. Такие МЭ разрабатывались для использования на отдельном компьютере, обеспечивая персональную защиту пользователя. К этому же времени относятся исследовательские разработки по созданию распределенных систем МЭ.

Далее рассмотрим основные элементы технологий построения межсетевых экранов.

4.1.1. Фильтрация пакетов

Сначала данная технология применялась на сетевом уровне, поэтому фильтрации подвергались только IP-адреса источника и назначения. В настоящее время анализ сетевого трафика при фильтрации пакетов проводится и на транспортном уровне.

Каждый IP-пакет исследуется на соответствие множеству правил. Эти правила устанавливают разрешение связи по содержанию заголовков сетевого и транспортного уровней модели TCP/IP, анализируется и направление передвижения пакета.

Фильтры пакетов контролируют:

- физический интерфейс, откуда пришел пакет;
- IPи (IP-адрес источника);
- IPн (IP-адрес назначения);
- тип транспортного уровня (TCP, UDP, ICMP);
- транспортные порты источника и назначения.

Схема архитектуры фильтра пакетов приведена на рис. 4.2.

Трансляция сетевых адресов. Межсетевой экран, фильтрующий пакеты, часто переадресует сетевые пакеты так, что выходной трафик осуществляется с другими адресами. Такая схема называется схемой трансляции адресов (NAT, Network Address Translation) и описана в RFC 1631.

Применение схемы NAT позволяет, во-первых, спрятать топологию и схему адресации доверенной сети, а во-вторых, использовать внутри организации пул IP-адресов меньшего размера. Схема функционирования трансляции адресов представлена на рис. 4.3.

Различают статическую и динамическую трансляцию адресов. При статической трансляции используется блок внешних адресов, которые назначаются запросам хостов локальной сети. При динамической трансляции все запросы хостов локальной сети имеют один и тот же адрес. Для динамической трансляции используется форма

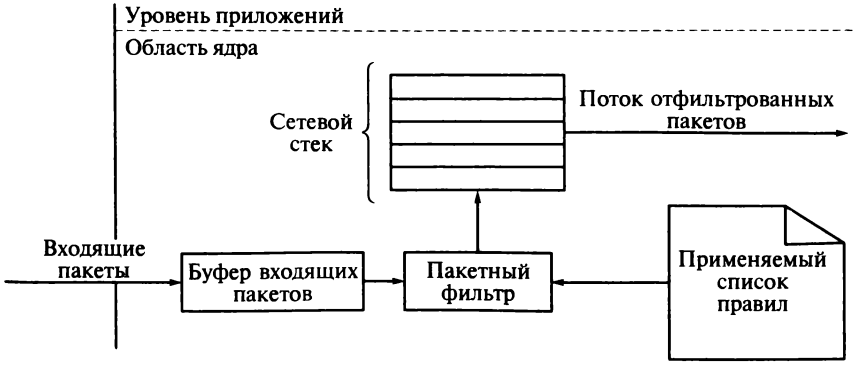


Рис. 4.2. Схема архитектуры фильтра пакетов

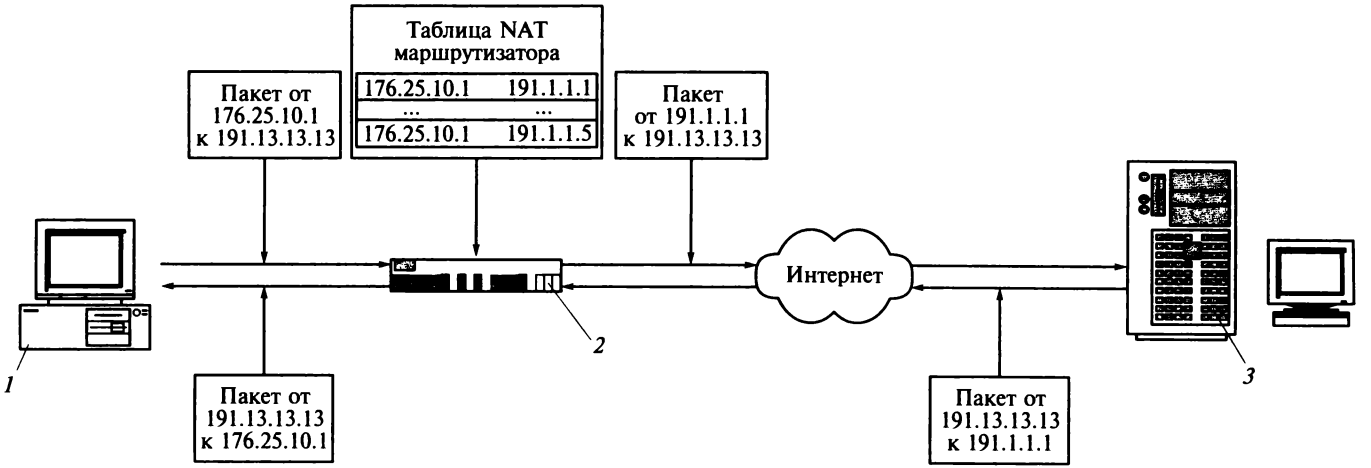


Рис. 4.3. Схема трансляции адресов:

1 — рабочая станция 176.25.10.1; *2* — маршрутизатор; *3* — сервер 191.13.13.13

(NAT Overloading), которая ставит в соответствие множеству адресов локальной сети единственный IP-адрес, используя различные номера портов (Port Address Translation, PAT).

Трансляция адресов, кроме скрытия внутренних адресов хостов локальной сети, выполняет важную функцию защиты. Если атакующий направит пакет на хост внутренней сети, то он будет отброшен, так как для него нет соответствующей строки в таблице NAT.

Процесс фильтрации пакетов. При фильтрации пакетов, если пакет удовлетворяет правилам, то он (в зависимости от направления от или к удаленному хосту) перемещается по сетевому стеку для дальнейшей обработки или передачи.

Все входные пакеты проверяются на соответствие заданным правилам фильтрации. Пакет уничтожается или разрешается для перемещения в сетевой стек для доставки. В такой архитектуре применяется ограниченное множество команд для анализа одного или нескольких сетевых протоколов, но она осуществляет анализ в пространстве ядра. Фильтр пакетов не разбирает, какой прикладной протокол будет использоваться. Правила содержат два списка: список запрещения (deny) и список разрешения (permit). Сетевой пакет проходит проверку на оба списка.

Общая схема исследования пакетов:

- если правило разрешает, то пакет допускается;
- если правило запрещает, то пакет удаляется;
- если ни одно правило не применено, то пакет удаляется.

Схема обработки пакетов при фильтрации приведена на рис. 4.4.

Технология фильтрации пакетов послужила основой создания различных средств защиты и реализована практически во всех типах маршрутизаторов. Основные достоинства и недостатки данной технологии приведены в табл. 4.1.

Списки контроля доступа. Для реализации процесса фильтрации пакетов применяются правила, называемые списками контроля доступа (Access Control List, ACL или просто Access List, AL).

В качестве примера рассмотрим реализацию фильтров в маршрутизаторах компании Cisco. Маршрутизатор Cisco выполняет фильтрацию пакетов посредством списка управления доступом, которые включены в Cisco Internetwork Operating System (IOS).

Список контроля доступа содержит перечень элементов в заголовках пакетов, которые будут проверяться. Маршрутизаторы Cisco определяют списки доступа как последовательный набор запрещающих и разрешающих условий. Каждый пакет проверяется на соответствие правилам списка. Если пакет соответствует правилу, то он отбрасывается (если это запрещающее правило) или передается далее (если это правило разрешающее). Если пакет соответствует правилу, то он уже не будет проверяться на соответствие остальным правилам. Поэтому порядок правил в списке доступа играет важную роль.

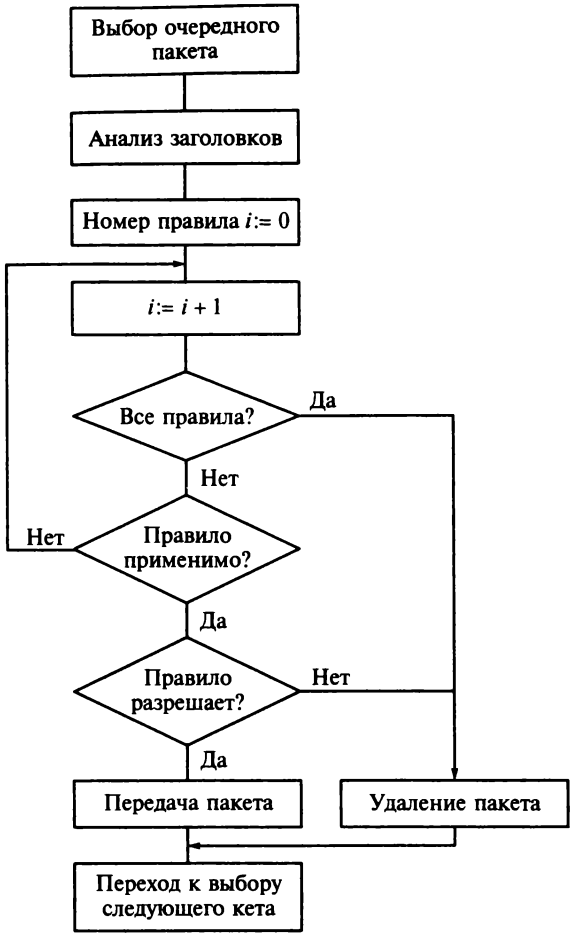


Рис. 4.4. Схема обработки пакетов при фильтрации

Существуют несколько типов списков контроля доступа. Простейшей фильтрации пакетов соответствует стандартный список управления доступом. При описании синтаксиса списков управления доступом Cisco значения, указанные в фигурных скобках, являются обязательными, а значения в квадратных скобках — необязательными. Жирным шрифтом будем выделять ключевые слова списка.

Синтаксис стандартного списка контроля управления доступом:

```
access-list list-number {permit/deny} source {mask}
[log]
```

Здесь *list-number* — номер данного списка доступа (для стандартных списков допустимы номера с 1 по 99). Ключевое слово

Таблица 4.1. **Достоинства и недостатки технологии фильтрации**

Преимущества	Недостатки
Быстрота работы (по сравнению с другими технологиями МЭ) МЭ может быть реализован аппаратно Не требуется конфигурирование хостов пользователя Схема NAT «прячет» внутренние IP-адреса	Не «понимает» прикладные протоколы Не может ограничить доступ подмножеству протоколов даже для основных служб (например, команд put, get FTP) Не отслеживает соединения (не содержит информацию о сеансе) Слабые возможности обработки информации внутри пакета Не может ограничить информацию с внутренних компьютеров к службам МЭ сервера Практически не имеет аудита Трудно тестировать правила (из-за сложности внутренних сетей, наличия различных служб)

permit разрешает прохождение пакета, удовлетворяющего условию, а ***deny*** — запрещает прохождение. При удалении пакета посылается сообщение ICMP о недостижимости назначения. Слово ***source*** определяет источник (хост или сеть), из которого послан пакет. Источник может быть определен IP-адресом или ключевым словом ***any***. Слово ***mask*** определяет биты маски для заданного адреса источника. По умолчанию маска равна 0.0.0.0, она определяет единственный IP-адрес. Чтобы не указывать маску, можно использовать слово ***host***, за которым следует его IP-адрес, например

```
access-list 15 permit host 192.168.123.45
```

или с указанием маски:

```
access-list 15 permit 192.168.123.45 0.0.0.0
```

Поскольку маска состоит из одних нулей, необходимо проверять каждый бит адреса. Для приведенных правил будет разрешаться только адрес 192.168.123.45 и запрещаться все другие адреса (вследствие подразумеваемого ***deny*** в конце списка).

Бит маски, установленный в единицу, означает, что данный бит не должен соответствовать соответствующему биту адреса. Таким образом, маска из всех единиц (255.255.255.255) означает, что не проверяются никакие биты адреса, т. е. разрешается весь трафик. Для об-

легчения использования случая, когда маска состоит из одних нулей или единиц, используется слово any. В качестве примера рассмотрим случай, когда необходимо запретить доступ хостам, адреса которых находятся в диапазоне от 192.168.10.32 до 192.168.10.63. Маска подсети для этого диапазона будет 255.255.255.224, а маска для фильтрации — 0.0.0.31.

Ключевое слово *log* вызывает регистрацию события, вызвавшего совпадение с утверждением правила.

Каждый маршрутизатор имеет, как минимум, два интерфейса. Интерфейс, связанный с внутренней сетью, обозначается Ethernet 0, а внешний интерфейс — Serial 0. При наличии большего числа интерфейсов они получают последовательно увеличивающиеся адреса, например Ethernet 0, Ethernet 1.

Приведем правила стандартного списка управления доступом для случая, когда только трафику хостов сети (192.168.20.0) разрешено проходить через маршрутизатор, за исключением хоста 192.168.20.13, хотя он и является хостом данной сети. Строка правила списка доступа, начинающаяся с восклицательного знака, означает комментарий:

```
access-list 25 deny host 192.168.20.13
! запрет доступа в защищаемую сеть данному хосту
access-list 25 permit 192.168.20.0 0.0.0.255
! разрешение доступа подсети класса C
```

Маршрутизаторы Cisco используют идеологию: «то, что не разрешено, запрещено», поэтому в каждом списке доступа последней строкой подразумевается *deny*. В приведенном примере будет запрещен весь неуказанный трафик. Правила списка контроля доступа обрабатываются последовательно. Поэтому важна последовательность написания правил. В нашем примере изменение порядка строк привело бы к тому, что хост 192.168.20.13 всегда бы имел доступ к внутренней сети, так как второе правило никогда бы не проверялось. Поэтому в списках контроля доступа всегда сначала пишутся утверждения с ключевым словом deny.

Стандартные списки контроля доступа Cisco выполняют функции простой фильтрации, т. е. являются межсетевыми экранами фильтрации пакетов.

4.1.2. Межсетевые экраны уровня соединения

Данные МЭ проверяют факт, что пакет является либо запросом на TCP-соединение, либо представляет данные, относящиеся к уже установленному соединению, либо относится к виртуальному соединению между двумя транспортными уровнями.

Для проверки соединения МЭ исследует каждое установленное соединение (проверяя законное «рукопожатие» для используемого

транспортного уровня — обычно TCP). Никакие пакеты не передаются до завершения рукопожатия. Для этого МЭ формирует таблицу действительных (установленных) соединений, которые включают в себя полную информацию о состоянии соединения и выполнении необходимой последовательности. Разрешается прохождение пакетов, информация в которых соответствует входу в таблицу виртуальных соединений. По окончании соединения соответствующий вход в таблицу удаляется. Схема функционирования МЭ данного вида представлена на рис. 4.5.

После установления соединения в соответствующей таблице (таблице состояний) обычно хранится следующая информация:

- идентификатор сеанса;
- состояние соединения (рукопожатие, установлено, закрыто);
- последовательная информация (последовательные номера пришедших байтов, состояния флагов и т. д.);
- IP-адрес источника и IP-адрес назначения;
- номера портов, участвующих в сеансе;
- физический интерфейс, куда прибыл пакет;
- физический интерфейс, куда передается пакет;
- временны́е метки начала открытия сеанса и т. д.

При функционировании такого МЭ должно обеспечиваться минимальное количество проверок, что реализуется посредством построения ограниченной формы состояний соединений. Для обеспечения дополнительных возможностей могут применяться дополнительные проверки (например, проверки соответствия данных, содержащихся в заголовке транспортного протокола, протоколу прикладного уровня).

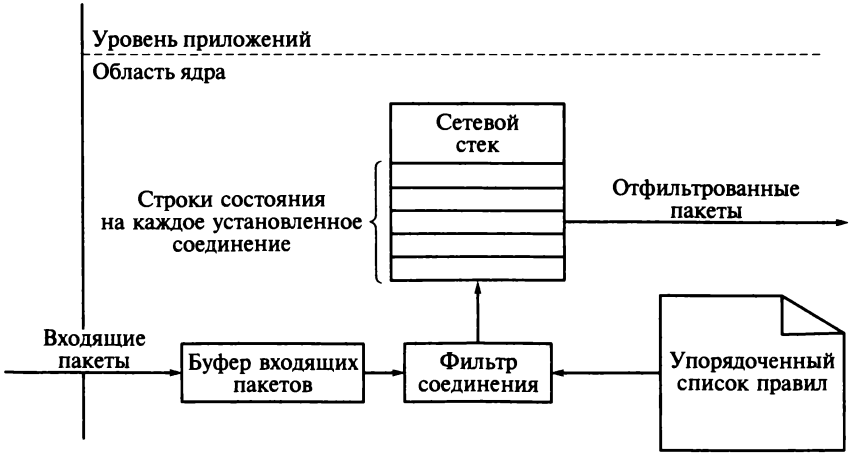


Рис. 4.5. Схема функционирования МЭ уровня соединения

Таблица 4.2. **Достоинства и недостатки технологии соединений**

Достоинства	Недостатки
Возможность запрещения соединений с определенными хостами При использовании NAT — скрытие внутренних IP-адресов	Не могут ограничить доступ протоколов, отличных от TCP Не осуществляют проверки для протоколов высших уровней Ограниченный аудит (слабая связь с высшими уровнями протоколов) Не позволяют дополнения функций — HTTP-кеширования ответов, фильтрации URL, аутентификацию Трудность тестирования правил

В данном случае также может использоваться NAT. Основные достоинства и недостатки данной технологии приведены в табл. 4.2.

Расширенные списки доступа Cisco позволяют фильтровать IP-адреса источника и назначения, дают возможность использования вложенного в IP-протокола (TCP, UDP, ICMP, BGP, IGRP) и порта назначения. В случае использования ICMP фильтруется код или тип сообщения, а в случае установления соединения TCP — установка флагов ACK и RST.

Синтаксис расширенного списка управления доступом:

access-list *list-number {**permit/deny**} protocol source s-mask [**operator** s-port] destination d-mask [**operator** d-port] [**precedence** значение] [**tos** значение] [**established**] [**log/log-input**]*

Расширенный список доступа должен иметь номер из диапазона 100... 199 или имя.

Если списку присваивается имя, то это задается специальной командой (*ip access-list extended* имя). В качестве протокола можно указывать как сокращение протокола (*ip*, *tcp*, *udp*, *icmp* и т.д.), так и номер протокола (от 1 до 255) в соответствии с номером, указываемым в заголовке IP-пакета (например, ICMP имеет номер 1, TCP — 6, UDP — 17). Ключевое слово *ip* в поле протокола означает все протоколы. Слово *s-mask* означает маску адреса источника, а *d-mask* — назначения. Слово *operator* применимо только для порта назначения (*d-port*). Допустимыми значениями поля *operator* являются:

- lt (less than) — меньше чем;
- gt (greater than) — больше чем;
- eq (equal to) — равно;

• neq (not equal to) — не равно;
• range — диапазон (в этом случае указываются два номера порта).

Списки контроля доступа разрешают вместо номеров портов использовать сокращения наименований служб, использующих эти порты. Поле precedence используется для фильтрации по уровням приоритетов (от 0 до 7), поле tos — для фильтрации типа обслуживания (от 0 до 15). Поле est (established) применяется только к протоколу TCP. Поле log указывает на регистрацию события, когда пакет удовлетворяет условиям списка доступа. Указание log-input добавляет к записям имя интерфейса, где получен соответствующий пакет.

Приведем примеры правил расширенного списка доступа и комментарии к ним:

```
access-list 141 permit icmp host 192.168.10.68
10.10.10.0 0.255.255.255
! разрешение хосту 192.168.10.68 посылать сообщения
ICMP
! любому хосту сети 10.10.10.0
access-list 141 permit tcp host 192.168.10.69 eq 734
10.10.10.0 255.255.255.255 range 10000 10010
! разрешение хосту 192.168.10.69 инициировать сеансы
TCP
! с порта 734 на любой порт в диапазоне с 10000 до
10010
! с любым хостом сети 10.10.10.0
access-list 141 permit udp host 192.168.10.90
10.10.10.0 255.255.255.255 eq ftp
! разрешение хосту 192.168.10.90 посылать файлы че-
рез TFTP
! (UDP порт 69) любому хосту сети 10.10.10.0
```

Расширенные списки управления доступом анализируют каждый пакет индивидуально и не имеют возможности определения того, что пакет является частью сеанса более высоких уровней.

Для соединения TCP используется ключевое слово *est*. При этом контролируется заголовок TCP на наличие флагов ACK и RST, но не проверяется то, что пакет действительно является частью установленного соединения. Поэтому расширенные списки доступа не защищают от поддельных пакетов TCP и не дают возможности фильтровать сеансы UDP.

4.1.3. Межсетевые экраны прикладного уровня

Данные МЭ оценивают сетевые пакеты на соответствие определенному прикладному уровню перед установкой соединения. Они

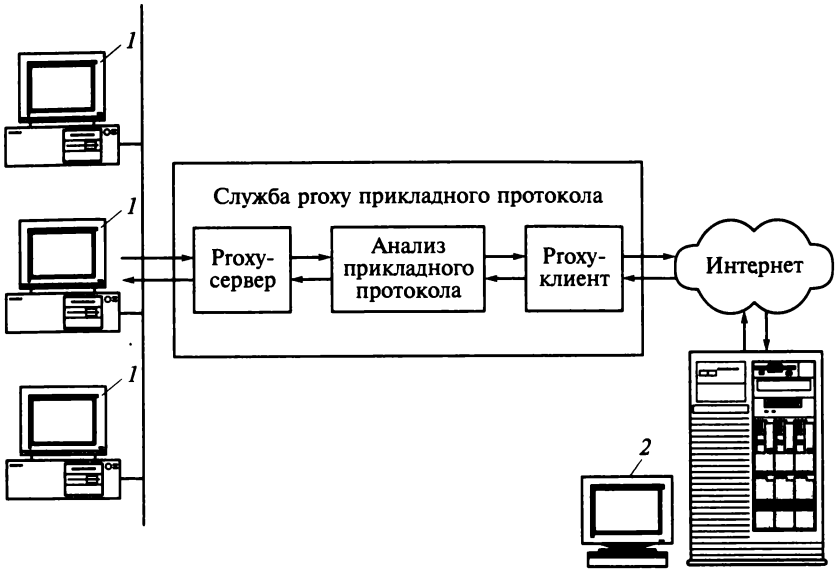


Рис. 4.6. Схема функционирования служб proxy:
1 — рабочая станция; *2* — сервер

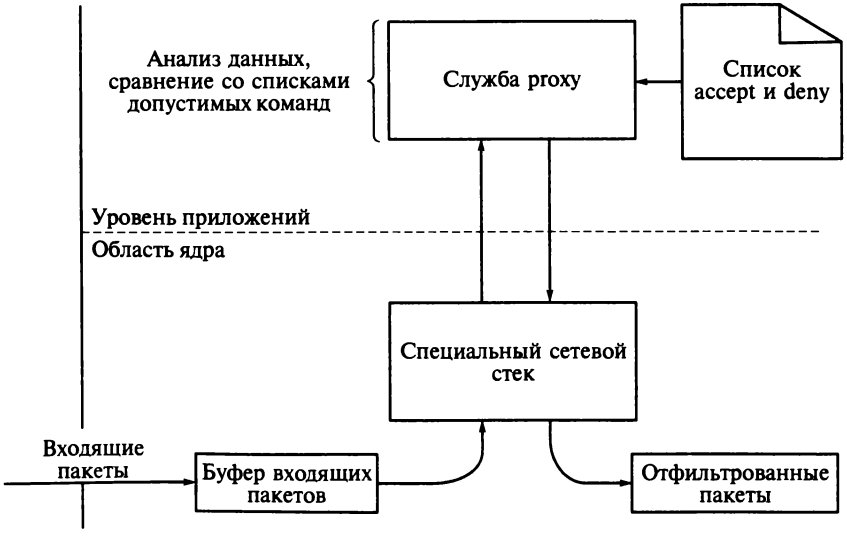


Рис. 4.7. Схема функционирования МЭ прикладного уровня

Таблица 4.3. **Достоинства и недостатки МЭ прикладного уровня**

Достоинства	Недостатки
Работа с протоколами высшего уровня (HTTP, FTP) Возможность хранения частичной информации о состоянии, полной информации состояния приложения и частичной информации о сеансе Возможность ограничения доступа к определенным сетевым службам Возможность оперирования с информацией данных пакета Запрещение прямого соединения с внешними серверами (внутренние имена спрятаны) Прозрачность proxy Возможность реализации дополнительных свойств (фильтрации URL, аутентификации, кэширования HTTP) Хороший аудит	Служба proxy требует замены сетевого стека на сервере МЭ Служба proxy слушает порт (как сетевой сервер, т.е. МЭ не может его использовать) Временная задержка (входной пакет обрабатывается дважды — приложением и proxy) Новый proxy должен быть добавлен для каждого контролируемого протокола Службы proxy обычно требуют модификации процедур клиентов Службы proxy уязвимы к ошибкам ОС и ПО прикладного уровня Не осуществляется проверка информации пакета, содержащейся в низших уровнях Служба proxy может требовать дополнительных паролей или процедур аутентификации

исследуют данные всех сетевых пакетов на прикладном уровне и устанавливают состояние полного (завершенного) соединения и последовательной информации. Кроме того, МЭ могут проверять другие параметры безопасности, которые содержатся внутри данных прикладного уровня (пароли, запросы служб).

Большинство МЭ прикладного уровня включают в себя специализированное прикладное ПО и службы proxy. Схема функционирования служб proxy представлена на рис. 4.6.

Каждая proxy-служба является специфичной для каждого протокола и может осуществлять усиленный контроль доступа, проверку данных, а также генерировать записи аудита. Службы proxy не позволяют прямого соединения пользователей с серверами, они прозрачны для пользователя и работают в прикладной области ОС.

Использование proxy позволяет проводить анализ множества команд для одного протокола и, что очень важно, проводить анализ содержания данных (фильтрацию URL, аутентификацию и т.д.). Схема функционирования МЭ прикладного уровня представлена на рис. 4.7.

Основные достоинства и недостатки МЭ данной технологии приведены в табл. 4.3.

4.1.4. Межсетевые экраны с динамической фильтрацией пакетов

Данные МЭ позволяют осуществлять модификацию базы правил «на лету» (on fly). Это реализуется для протокола UDP. В этом случае МЭ осуществляет ассоциацию всех UDP пакетов, которые пересекают периметр безопасности через виртуальное соединение. Если генерируется пакет ответа и передается источнику запроса, то устанавливается виртуальное соединение и пакету разрешается пересечь сервер МЭ. Информация, ассоциированная с виртуальным состоянием, запоминается на краткий промежуток времени, поэтому если пакет ответа не получен, то соединение считается закрытым.

Схема функционирования МЭ с динамической фильтрацией приведена на рис. 4.8.

Главной особенностью данной схемы является наличие двух групп правил. Одна — статическая, другая — динамическая, изменяемая в ходе работы МЭ. Основные достоинства и недостатки данного вида МЭ приведены в табл. 4.4.

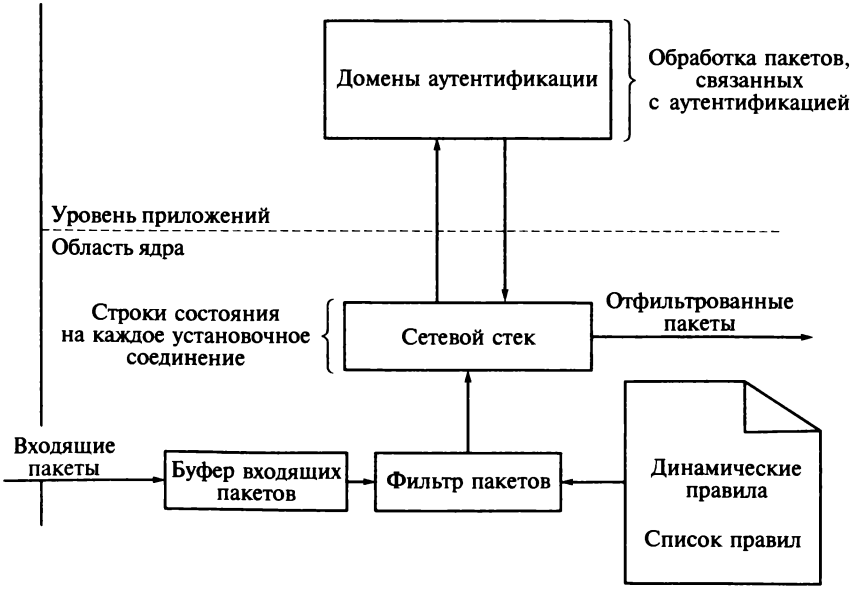


Рис. 4.8. Схема функционирования МЭ с динамической фильтрацией

Таблица 4.4. **Достоинства и недостатки МЭ с динамической фильтрацией**

Достоинства	Недостатки
Не позволяет непрошеным пакетам UDP войти во внутреннюю сеть Если запрос пакета UDP приходит из внутренней сети и направлен на не доверенный хост, то сервер МЭ позволяет появление ответного пакета, доставляемого хосту — инициатору запроса. Пакет ответа должен содержать IP$_H$ (соответствующий запросу) и номер порта (соответствующий запросу) и иметь соответствующий тип протокола транспортного уровня Динамический фильтр может использоваться для поддержки ограниченного множества команд ICMP	Не «понимает» прикладные протоколы Не может ограничить доступ подмножеству протоколов даже для основных служб Не отслеживает соединения Слабые возможности обработки информации внутри пакета Не может ограничить информацию с внутренних компьютеров к службам МЭ сервера Практически не имеет аудита Трудно тестировать правила (из-за сложности внутренних сетей, наличия различных служб)

Технология динамической фильтрации пакетов используется не только для протокола UDP и опирающихся на него прикладных протоколов, поэтому ее можно назвать технологией установления виртуального соединения или виртуального сеанса.

Динамические списки контроля доступа Cisco (Lock-and-Key по терминологии Cisco) позволяют автоматически открывать вход в существующий на маршрутизаторе список доступа, который будет фильтровать входящий трафик от аутентифицированного пользователя. Сначала пользователь открывает сеанс telnet с маршрутизатором для проведения аутентификации (если это разрешено соответствующим расширенным списком доступа). Маршрутизатор в сеансе запрашивает имя и пароль пользователя (в соответствии с установками администратора, причем для аутентификации могут использоваться сервера доступа, например, TACACS+ или RADIUS). При успешном прохождении аутентификации сеанс telnet закрывается и создается временный вход в динамический список доступа. Такой динамический вход разрешает трафик между хостом (по IP-адресу) пользователя и определенным хостом защищаемой сети. Динамический список доступа удаляется по истечении заданного промежутка времени или по команде администратора.

Синтаксис динамического списка контроля доступа:

access-list *list-number [**dynamic** имя_списка [**timeout** минуты]] {**deny/permit**} protocol source s-mask destination d-mask [**precedence** значение] [**tos** значение] [**established**] [**log/log-input**]*

Номер списка доступа должен быть в диапазоне от 100 до 199. Слово *dynamic* обозначает данный вход как часть динамического списка с именем *имя_списка*. Если необходимо иметь несколько входов, переключающихся в одно и то же время, то они должны иметь одно имя. Слово *timeout* задает промежуток времени существования динамических входов.

Любой вход в списке доступа, не помеченный как динамический, будет фильтроваться как основной расширенный список доступа. Для каждого существующего на маршрутизаторе списка доступа может быть установлен только один динамический список. Чтобы открыть временный вход, определенный в списке доступа, пользователь должен ввести команду (в ответ на приглашение):

access-enable *[**host**] [**timeout** минуты],*

где *access-enable* — сообщение маршрутизатору об активации временного входа в динамическом списке; *host* — адрес хоста, трафику которого после аутентификации разрешено проходить через динамический список; *timeout* — значение «времени простоя». Если трафик будет отсутствовать в течение указанного промежутка времени, то временный вход удаляется.

Пусть внешнему пользователю (10.10.10.10) необходимо провести сеанс FTP длительностью до 60 мин с хостом 192.168.100.1 сети, защищаемой маршрутизатором. Внешний интерфейс маршрутизатора имеет адрес 172.16.20.2. Тогда соответствующий список доступа может иметь следующий вид:

```
access-list 123 dynamic remoteuser timeout 60 permit tcp
any host 192.168.100.1 eq ftp
access-list 123 permit tcp any host 172.16.20.2 eq telnet
```

В этом случае разрешен только telnet доступ к маршрутизатору. Любой, кто пройдет аутентификацию, получит доступ к хосту внутренней сети. Если пользователь, прошедший аутентификацию, наберет команду access-enable host 10.10.10.10, то это добавит его адрес в список доступа. В этом случае реальный список 123 в маршрутизаторе будет содержать следующие правила:

```
dynamic remote timeout 60 permit tcp any host 192.168.100.1 eq ftp
```

```
permit host 10.10.10.10 host 192.168.100.1 eq ftp
permit tcp any host 172.16.20.2 eq telnet
```

Эти правила показывают, что пользователь прошел аутентификацию и конкретному адресу разрешен доступ FTP к хосту внутренней сети.

Временный вход не удаляется, когда пользователь закончит сеанс. Он удаляется, если истек промежуток времени, указанный параметром timeout в списке доступа, или при наличии времени простоя, или по команде администратора.

Использование списков Lock-and-Key не защищает, если злоумышленник использует поддельные адреса. Достоинством использования этого списка управления доступа является открытие доступа на заданный промежуток времени.

4.1.5. Межсетевые экраны инспекции состояний

Технология инспекции состояний (stateful inspection) осуществляет анализ пакетов на трех высших уровнях. Этот подход используется многими разработчиками, но, поскольку наименование запатентовано компанией Check Point, они вынуждены присваивать ему различные наименования (кроме stateful inspection используются expert inspection, smart filtering, adaptive screening, multilevel inspection и др.).

Устройство инспекции состояний осуществляет анализ пакетов и формирование данных о «состоянии виртуального соединения». Соединение может находиться в состоянии установки, передачи или отключения. В каждом из этих состояний имеется возможность интерпретировать коммуникационные данные определенным способом.

Вся информация, связанная с состоянием данного виртуального соединения, хранится в таблице динамических состояний, с помощью которой оценивается дальнейший обмен в рамках этого виртуального соединения, т. е. осуществляется контроль последовательности пакетов на различных уровнях. Схема фильтрации при инспекции состояний приведена на рис. 4.9.

Отслеживание информации прикладного уровня позволяет учитывать поведение нестандартных протоколов (например, FTP или H.323). Каждый разработчик межсетевого экрана использует свою реализацию для построения таблицы состояний.

Межсетевой экран IP-tables является свободно распространяемым продуктом для Linux. При отслеживании состояния соединение регистрируется в основном на основании информации о протоколе.

Администратор имеет возможность создать правила, определяющие, какие протоколы или определенные виды трафика необходимо отслеживать. Когда соединение начинается с использованием отсле-

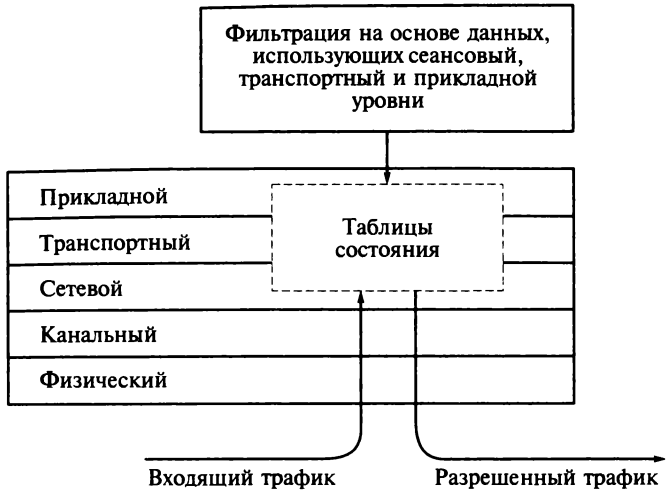


Рис. 4.9. Схема фильтрации при инспекции состояний

живаемого протокола, IP-tables добавляет записи в таблицу состояний всего соединения. Запись в таблице состояний включает в себя следующую информацию:

- протокол, используемый для соединения;
- IP-адреса источника и назначения;
- номера портов источника и назначения;
- листинг с обращенными адресами и номерами портов (для контроля возвращаемого трафика);
- время, по истечению которого соединение будет удалено;
- состояние TCP-соединения (только для TCP);
- состояние отслеживаемого соединения.

Пример записи в таблице состояний для IP-tables:

```
tcp 6 93 SYN_SENT src=192.168.1.1 dst=192.168.200.200
sport=1054 dport=21 [UNREPLIED]
src=192.168.200.200 dst=192.168.1.1 sport=21
dport=1054 use=1
```

В первой строке указан протокол и его числовое обозначение (6 для TCP), следующее значение (93) отражает время, по истечению которого запись будет автоматически удалена из таблицы состояний. Далее указано состояние соединения TCP (послан SYN). Указаны номера адресов и портов. Межсетевой экран видит это соединение как [UNREPLIED], так как это начало установления соединения. В третьей строке указана реверсная (обращенная) строка для разрешения обратного трафика.

После установления соединения запись в таблице состояний изменится на следующую:

```
tcp 6 41294 ESTABLISHED src=192.168.1.1
dst=192.168.200.200
sport=1054 dport=21 src=192.168.200.200 dst=192.168.1.1
[ASSURED] use=1
```

Маркер [UNREPLIED] удален после получения первого обратного пакета, а по установлению соединения помещен маркер [ASSURED] и увеличена величина тайм-аута (41 294).

Межсетевой экран Check Point FireWall-1 представляет собой один из наиболее популярных экранов инспекции состояний. Он является программным продуктом и может быть установлен на различные платформы, включая Windows NT/2000, Solaris и Red Hat Linux. Этот межсетевой экран использует таблицу состояний для основного отслеживания соединения на уровне протокола и модуль INSPECT для отслеживания более углубленных правил, включая трафик на прикладном уровне и поведение нестандартного протокола.

При принятии решения о разрешении прохождения пакета межсетевой экран проверит его последовательно по следующим структурам данных:

1) таблица состояний — зарегистрировано ли соединение для данного входящего пакета (если да, то пакет передается без дальнейшей проверки);

2) политика безопасности — если правило разрешает прохождение пакета, то пакет будет передан, а для его сеанса соединения будет добавлена запись в таблицу состояний.

Для задания правил администратору предоставляется графический интерфейс, в котором содержатся следующие опции: Source, Destination, Service, Action, Track, Install On и Time. В качестве примера приведем фрагмент листинга таблицы состояний Check Point FireWall-1, декодированный сценарием Perl — fwtable.pl, находящимся на странице Ланса Шпицнера (http://www.enteract.com/~lspitz/fwtable.txt). Для удобства чтения расположим строки таблицы двумя частями:

Src-IP	Src_Prt	Dst_Ip	Dst_Prt
192.168.1.202	1783	192.168.1.10	137
192.168.1.202	1885	192.168.1.10	80

Ip_prot	Kbuf	Type	Flags	Timeout
17	0	16386	ffffff00	18/40
6	0	28673	ffffff00	43/50

Настраиваемые proxy (adaptive proxy) — один из вариантов данной технологии. В этом случае начальное исследование состояния проводится на прикладном уровне. После формирования состояния в таблицу правил добавляется стандартизованная группа правил, соответствующая данному соединению и установленному режиму

безопасности. Если пакет удовлетворяет требованиям правил для данного виртуального соединения, то он передается через сетевой уровень, не затрагивая прикладной уровень. Это позволяет повысить быстродействие МЭ.

Примером подхода, в котором используются элементы технологии инспекции состояний, являются рефлексивные списки контроля доступа и списки контроля по содержимому Cisco.

Рефлексивные списки контроля доступа Cisco позволяют автоматически создавать временные входы при начале сеанса обмена. Эти списки включаются в существующий расширенный список, когда новый сеанс (TCP или UDP) инициализируется из внутренней (защищаемой) сети.

При включении рефлексивный список доступа автоматически генерирует новый временный вход, который будет разрешать трафику вход во внутреннюю сеть только тогда, когда трафик является частью сеанса. Рефлексивный список не содержит подразумеваемого ключевого слова *deny all* в конце, так как после обработки рефлексивного списка (если пакет не соответствовал правилам рефлексивного списка) маршрутизатор продолжит работу с остальной частью расширенного списка доступа.

Предположим, что пользователь внутренней сети, которому это разрешено расширенным списком, послал TCP пакет начала соединения. В этом случае создается временный вход рефлексивного списка доступа, который будет применяться к входящему во внутреннюю сеть трафику. Такой временный вход имеет следующие характеристики:

- всегда является входом с ключевым словом *permit*;
- определяет тот же протокол (в нашем случае TCP), что и пакет-инициатор сеанса;
- определяет адреса и порты источника и назначения, соответствующие пакету-инициатору (для протокола ICMP указываются типы сообщений);
- входящий трафик проверяется этим временным входом, пока вход существует;
- вход удаляется после прохождения последнего пакета сеанса, прошедшего через интерфейс маршрутизатора;
- если за заданный промежуток времени (заданное время простоя) нет пакетов, относящихся к сеансу, то вход удаляется.

Для сеансов TCP вход удаляется через 5 с после обнаружения двух пакетов с установленным флагом FIN или немедленно, если пакет содержит установленный флаг RST.

Для UDP и других протоколов, не ориентированных на соединение, завершение сеанса осуществляется по критерию времени простоя.

Для определения рефлексивного списка контроля доступа используются следующие команды:

```
permit protocol source s-mask destination d-mask
reflect reflect-name [timeout секунды]
```

Для вставки рефлексивного списка используется следующая команда:

```
ip access-list extended {list-name} evaluate {reflect-name}
```

Рассмотрим пример, в котором будем давать пояснения в строке, следующей за строкой списка доступа (знак «!» для маршрутизатора Cisco означает строку комментария). Маршрутизатору задаются команды применения списков к различным интерфейсам (*in* — с внешней стороны, входящий трафик, *out* — с внутренней, исходящий трафик):

```
interface Serial 0
! интерфейс маршрутизатора со стороны внешнего мира —
! Serial 0
description Access to the Internet via this
interface
! ключевое слово description означает комментарий
ip access-group infilters in
ip access-group outfilters out
! infilters и outfilters — имена списков доступа
! ключевое слово access-group позволяет задать спи-
ски
! на интерфейсы
ip reflexive-list timeout 120
! задание времени для всех рефлексивных списков
! по умолчанию — 300 секунд
ip access-list extended outfilters permit tcp any
any reflect tcptraffic
! указан рефлексивный список с именем tcptraffic для
! протокола TCP в расширенном списке outfilters
ip access-list extended infilters permit
permit bgp any any
deny icmp any any
evaluate tcptraffic
! разрешение использования протокола BGP и запрет
ICMP для
! входящего в маршрутизатор трафика
```

Аналогичным образом выглядят списки доступа для внутреннего интерфейса маршрутизатора (Ethernet 0):

```
interface Ethernet 0
description Access from the Internet to our network
```

```
via this interface
ip access-group infilters in
ip access-group outfilters out
ip reflexive-list timeout 120
ip access-list extended outfilters
permit bgp any any
deny icmp any any
evaluate tcptraffic
ip access-list extended infilters permit tcp any any
reflect tcptraffic
```

Рефлексивные списки способны контролировать только единственный канал приложения (такой как telnet), использующего единственный статический порт во время сеанса. Для устранения этого недостатка используются списки контроля доступа по содержимому.

В идеологию построения *списков контроля доступа по содержимому* компанией Cisco введено свойство CBAC (Context-Based Access Control). Это свойство позволяет контролировать и управлять различными типами информации прикладного уровня (инспекция состояний — stateful inspection). Когда определенный трафик выходит из внутренней сети, создается специальный вход в существующий список контроля доступа, который будет разрешать возвращаемый трафик. При этом соответствующие данные заносятся в таблицу состояний. В число этих данных входят IP-адреса, номера портов и т. д. Эта таблица состояний используется CBAC для создания временных входов в списках доступа для возвращаемого трафика. Если в рефлексивных списках фильтруется информация сетевого и транспортного уровней, то CBAC контролирует сетевой и транспортный уровни совместно с информацией прикладного уровня. Инспекция информации прикладного уровня осуществляется для подтверждения того, что входящему трафику разрешено проходить маршрутизатор. CBAC функционирует для следующих протоколов: TCP, UDP, CU-See Me, FTP, H.323 (для NetMeeting и ProShare), Java-апплеты (передаваемые через HTTP), Unix r-commands (таких как rlogin, rexec, rsh), RealAudio, RPC (Sun RPC), SMTP, SQL*Net, StreamWorks, TFTP и VDOLive.

При закрытии соединения вход в таблицу состояний удаляется вместе с соответствующим временным входом в список контроля доступа.

Для конфигурирования CBAC необходимо указать протокол, интерфейс маршрутизатора, направление трафика и список контроля доступа исходящего трафика.

Для определения CBAC используется следующий синтаксис:

ip inspect name *inspection-name protocol [**alert** {on/off} {**audit-trail** {on/off}] {**timeout** seconds}*

Ключевое слово *alert* позволяет послать сообщение на сервер регистрации, когда возникает нарушение в контролируемом приложении. Ключевое слово *audit-trail* разрешает запись соединений, используемых для защищаемого приложения (регистрируются следующие данные: адреса, порты, число переданных байт и т.д.).

Для применения CBAC необходимо определить трафик, которому разрешено выходить в Интернет. Далее необходимо создать список фильтрации входящего трафика, которая будет осуществляться независимо от CBAC.

Непосредственно для использования CBAC необходимо задать временны́е промежутки и значения порогов, которые будут использоваться для определения продолжительности неактивных сеансов связи перед их удалением. Для протокола TCP — это число полуоткрытых сеансов, а для UDP — промежуток времени до прихода возвращаемого трафика (ответа). Соответствующими командами CBAC являются:

```
ip inspect tcp synwait-time seconds
! Время ожидания ответа на SYN, по умолчанию —
! 30 секунд
ip inspect tcp finwait-time seconds
! Время ожидания после получения FIN, по умолчанию —
! 5 секунд
ip inspect tcp idle-time seconds
! Время простоя, по умолчанию — 3600 секунд, т.е.
! 1 час
ip inspect udp idle-time seconds
! Время простоя, по умолчанию — 5 секунд
ip inspect max-incomplete low number
! Нижний порог количества полуоткрытых соединений,
! по умолчанию — 400
ip inspect max-incomplete high number
! Верхний порог количества полуоткрытых соединений,
! по умолчанию — 500
ip inspect one-minute low number
! Нижний порог количества полуоткрытых соединений за
! одну минуту, по умолчанию — 400
ip inspect one-minute high number
! Верхний порог количества полуоткрытых соединений
! за одну минуту, по умолчанию — 500
ip inspect tcp max-incomplete host number block-
time
seconds
! Максимально допустимое число полуоткрытых
! соединений с одним хостом
```

В качестве примера рассмотрим случай, когда пользователям сети 192.168.130.0 разрешается работать с Интернетом, в том числе загружать файлы и использовать RealAudio (список контроля доступа называется *company*):

```
ip inspect name company ftp timeout 3600
ip inspect name company realaudio timeout 3600
ip inspect name company ftp timeout 3600
ip inspect name company udp timeout 15
ip inspect name company tcp timeout 3600
! Перечислены протоколы, которые разрешены
! пользователям
! локальной сети и которые будут контролироваться
! списком контроля доступа по содержимому (CBAC)
interface ethernet 0
  ip address 192.168.130.0 255.255.255.0
  ip access-group outbound in
  ip inspect company in
! Применен список доступа outbound к входящему
! трафику
! на интерфейс ethernet 0 (исходящий из локальной
! сети)
interface serial 0
ip address 192.168.130.0 255.255.255.0
  ip access-group inbound in
! Применен список доступа inbound к входящему
! трафику на интерфейс serial 0 (входящий
! в локальную сеть)
ip access-list extended outbound
  permit ip 10.150.130.0 0.0.0.255 any
  deny ip any any log
! Разрешены только пакеты, которые исходят из
! локальной сети
! и имеют адрес источника из внутренней сети, что не
! даст
! инициировать некоторые атаки из локальной сети
ip access-list extended inbound
  permit icmp any 10.150.130.0 0.0.0.255 echo-
reply
  permit icmp any 10.150.130.0 0.0.0.255
traceroute
  permit icmp any 10.150.130.0 0.0.0.255 time-
exceeded
  permit icmp any 10.150.130.0 0.0.0.255
unreachable
```

```
   deny ip any any log
! Разрешен вход в локальную сеть определенным ICMP
! сообщениям
! для обеспечения работы пользователей
```

Таким образом, рефлексивные списки доступа Cisco представляют элемент использования технологии инспекции состояний.

4.1.6. Межсетевые экраны уровня ядра

При оценке практических реализаций МЭ в качестве одного из главных параметров выступает быстродействие. Сложность проверок, осуществляемых МЭ, обычно приводит к снижению его производительности. Для устранения этого существенного недостатка была разработана технология МЭ, функционирующего на уровне ядра (Kernel Proxy). Рассмотрим подробнее особенности данной технологии на примере МЭ Cisco Centry Firewall, который впервые использовал данную технологию (основное внимание будем уделять использованию элементов уже рассмотренных технологий).

Межсетевой экран Cisco Centri включает в себя основные достоинства предыдущих архитектур (скорость обработки в ядре, зависимость от сессии для каждого уровня протокола и т.д.). Этот МЭ разработан с использованием технологии автономных агентов, что позволяет легко добавлять новых агентов для решения новых задач. Схема функционирования МЭ уровня ядра приведена на рис. 4.10.

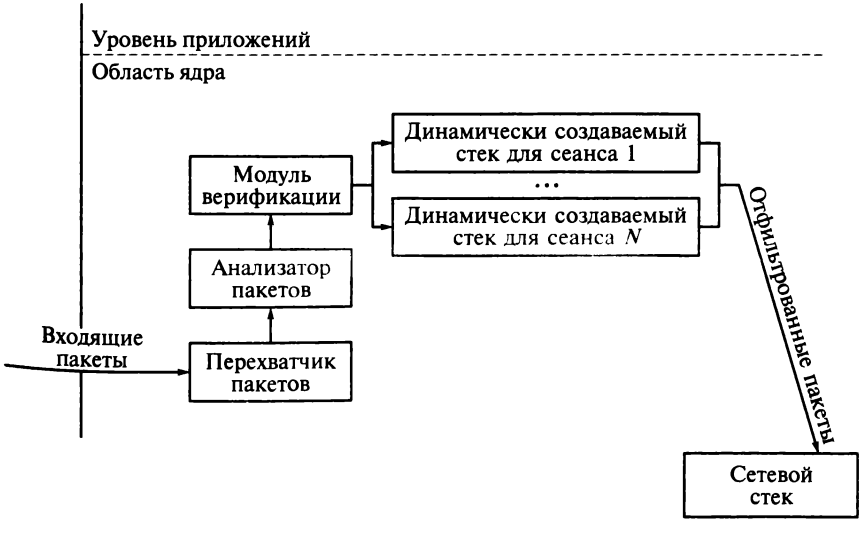


Рис. 4.10. Схема функционирования МЭ уровня ядра

Подсистема безопасности данного МЭ использует многие элементы из рассмотренных технологий МЭ. Подсистема МЭ Cisco Centri включает в себя следующие основные модули:

- ядро безопасности;
- модуль управления хостом;
- модуль управления каналами связи МЭ;
- агент регистрации входов;
- агент аутентификации.

Основным модулем является ядро безопасности. Ядро анализирует каждый входящий и исходящий пакет, проходящий через сервер МЭ, и применяет к каждому пакету заданную реализацию установленной политики безопасности. Ядро безопасности оперирует внутри ядра Windows NT, что позволяет обеспечить высокую производительность МЭ.

Основываясь на политике безопасности (применительно к каждой сетевой карте), модуль управления управляет созданием соединения для соответствующей карты. Каждый сетевой пакет анализируется на принадлежность к существующему соединению. Если такое соединение для пакета есть, то пакет передается в стек proxy существующего соединения. Если нет — проверяется модулем управления на наличие политики соединения. В результате пакет или удаляется, или создается динамический стек для нового соединения, и пакет передается в него.

Ядро безопасности, в свою очередь, содержит четыре компонента, составляющих суть данной технологии.

Перехватчик пакетов перехватывает пакеты, поступающие на МЭ, и передает их в модуль верификации. Перехватчик располагается между естественным сетевым стеком Windows и драйверами сетевых адаптеров. Для перехватчика стек Windows является внешним стеком, поэтому перехватчик рассматривает все пакеты до их поступления в стек собственно Windows.

Захватив пакет, перехватчик передает его в *анализатор*, который по информации заголовка пакета подготавливает пакет и соответствующие данные для дальнейшей работы модуля верификации.

Модуль верификации безопасности применяет заданную политику безопасности (загружаемую в ядро административным агентом), инициализирует и отслеживает сеансы всех разрешенных соединений. Таким образом, модуль верификации на основании пакета и подготовленных анализатором данных для каждого пакета выполняет одно их трех возможных действий:

- отбрасывает пакет (исключает из дальнейшей обработки);
- устанавливает, что пакет принадлежит существующему соединению, и передает пакет соответствующему proxy;
- устанавливает (если пакету разрешено) новое соединение с созданием соответствующего динамического стека proxy.

Механизмы proxy-ядра используют *динамические стеки*, зависящие от соответствующего протокола соединения. В первой версии МЭ были реализованы восемь типов динамических стеков.

Рассмотрим виды proxy в МЭ уровня ядра и выполняемые в их рамках проверки.

Для IP:

- проверка адресов источника и назначения — проверяется, что данному адресу источника разрешено взаимодействовать с соответствующим адресом назначения;
- защита от атаки Ping of Death — проверяется, что каждый пакет не превышает максимально возможную длину. Если длина пакета больше максимальной, то пакет уничтожается и генерируется запись в журнал;
- защита от атаки IP Spoof — при установлении нового соединения проверяется источник по статическим таблицам маршрутов, связанным с сетевыми картами. Если пришедший пакет не соответствует установленной таблице, то он уничтожается и генерируется запись в журнале.

Для ICMP:

- проверяется соответствие типа пришедшего пакета ICMP заданным правилам. Если данный тип не разрешен, то пакет уничтожается и генерируется запись в журнале.

Для TCP:

- проверка портов — проверяется, что запрос на инициализацию соединения направлен на разрешенный порт;
- защита от атаки SYN Flood — данная проверка оперирует с заранее заданным счетчиком числа полуоткрытых соединений. Значение счетчика может устанавливаться автоматически в зависимости от реальной физической памяти стека. При превышении значения этого счетчика проводится анализ процентного заполнения стека (для полуоткрытых соединений) и адресов их источников. Для тех адресов, которые были получены ранее, полуоткрытые соединения закрываются;
- функция NAT — выполнение функций трансляции адресов.

Для UDP:

- проверяется, что запросы на инициализацию служб направлены на разрешенные порты.

Для HTTP:

- аутентификация пользователя — инициируется передачей соответствующих данных запроса на соединение агенту аутентификации;
- фильтрация HTTP — проверка осуществляется в соответствии со списком файлов и сайтов, которые разрешено использовать в службе HTTP. В этом списке могут быть указаны адреса, имена доменов или типы файлов (по расширению, например, .class, .ocx, .alx);
- контроль разрешенных действий — проверяются все действия, ассоциированные с HTTP, на наличие соответствующего разреше-

ния для пользователя (например, поместить объект на HTTP сервер, получить объект с HTTP-сервера);

• фильтрация HTML — проверяется содержимое пакетов прикладного уровня. В ходе этой проверки модифицируются, если необходимо, данные HTML-документов, передающихся во время HTTP сеанса: фильтрация ActiveX (удаляются теги <OBJECT>), фильтрация Java апплетов (удаляются теги <APPLET>), фильтрация JavaScript и VBScript (удаляются теги <SCRIPT>).

Для FTP:

• аутентификация пользователя — инициируется передачей соответствующих данных запроса на соединение агенту аутентификации;

• поддержка режима «непрозрачного» proxy — данный режим выполняется в случае, когда адрес в пакете является адресом самого МЭ. Пользователи могут соединиться с сервером МЭ и, используя команды FTP, с другими серверами. Этот режим предназначен для работы внешних пользователей с серверами компании;

• контроль разрешенных действий — осуществляется проверка всех возможных действий, ассоциированных с FTP, наличие разрешений пользователя на использование соответствующей службы, предоставляемой протоколом: чтение объекта с FTP-сервера, запись объекта на FTP-сервер, удаление объекта с FTP-сервера, перемещение по директориям.

Для Telnet:

• аутентификация пользователя — инициируется передачей соответствующих данных запроса на соединение агенту аутентификации;

• поддержка режима «непрозрачного» proxy — выполняется в случае, когда адрес в пакете является адресом самого МЭ. Пользователи могут соединиться с сервером МЭ и затем, используя команды Telnet, с другими серверами. Этот режим предназначен для работы внешних пользователей с серверами компании;

• контроль портов — проверяется, что запросы на инициализацию служб направлены на разрешенные Telnet-порты.

Для SMTP:

• противодействие атаке SMTP Flood — позволяет удалять всю почту от определенного адреса или сети, направленную на защищаемую сеть;

• разрешение знаков маршрутизации — показывает, разрешено ли пользовательской части заголовка сообщения содержать знаки маршрутизации: !, [,], : и %;

• ограничение числа знаков At (@) — показывает, сколько знаков @ может содержать путь, указанный пользователем;

• разрешение команды Verify — проверяется, разрешено ли использование команды VRFY при прохождении МЭ на пути к серверу SMTP;

• разрешение команды Expand — проверяется, разрешено ли использование команды EXPN при прохождении МЭ на пути к серверу SMTP;

• ограничение числа получателей — позволяет установить количество абонентов, которым адресуется одно сообщение;

• ограничение длины сообщения — позволяет устанавливать максимальную разрешенную для передачи длину сообщения (в байтах).

В данном МЭ применяются правила фильтрации. Существуют статические и динамические правила фильтрации для соединений, организован контроль на прикладном уровне и т.д. Кроме того, весь внутренний трафик между базой знаний МЭ и агентами шифруется с помощью Microsoft Crypto API.

Хотя сам Cisco Centri снят с производства, его идеи нашли широкое применение в разработках разных производителей. Большинство современных МЭ оперируют на уровне ядра операционной системы, в качестве которой выступают не операционные системы общего пользования, а специально разрабатываемые операционные системы.

4.1.7. Персональные межсетевые экраны

Рассмотренные виды МЭ, особенно МЭ прикладного уровня и уровня ядра, являются чрезвычайно сложными и дорогостоящими продуктами, поэтому для защиты компьютеров отдельных пользователей стали разрабатываться персональные МЭ. Персональные МЭ (их называют также встроенными, embedded) являются программными продуктами, которые располагаются внутри компьютера на низшем уровне ОС — между сетевыми платами и всеми протокольными стеками (TCP/IP, NetBEUI, IPX и т.д. для Windows).

Персональные МЭ обычно выполняют две основные защитные функции: защиту от внешних атак и защиту от атак со стороны данного компьютера.

В качестве примера кратко рассмотрим двух представителей семейства персональных межсетевых экранов: Windows XP Firewall и Tiny Firewall.

Windows XP Firewall. Пакет обновлений Service Pack 2 для Windows XP включает в себя Windows Firewall, являющийся персональным межсетевым экраном, использующим технологию контроля состояний.

Этот межсетевой экран устанавливает два множества конфигураций, называемых профилями. Один профиль (Domain Profile) используется, когда компьютер находится в домене, а другой — если компьютер не соединен с доменом (Standard Profile). По умолчанию межсетевой экран использует Standard Profile. Для каждого профиля создается список исключений.

Параметры МЭ, заданные по умолчанию, в большинстве случаев не требуют изменений и ориентированы на использование обычного клиентского доступа в Интернет (просмотр Web-страниц, работа с электронной почтой). При этом все не запрашиваемые пользователем входящие подключения отклоняются. Межсетевой экран блокирует весь входящий трафик, который не является ответом на запрошенный трафик, и трафик, который не был определен как разрешенный.

Через простой интерфейс пользователь может задать основные режимы, функции и параметры работы. Кратко перечислим основные установки:

- функционирование в одном из трех режимов (On, On with No Exceptions, Off). В режиме On (включен) блокируется весь входящий трафик, за исключением того, который указан в списке исключений. В режиме On with No Exception (включен и без исключений) блокируется весь трафик, даже если он указан в списке исключений. В режиме Off (выключен) межсетевой экран не блокирует ничего;
- отключение сообщений (уведомлений) пользователю, которые выводятся, в случае если программа, не включенная в список исключений, пытается добавить себя или свой трафик в список исключений;
- блокировка входящих ответов на широковещательные запросы. По умолчанию межсетевой экран позволяет посылку одиночных пакетов на порт в течение трех секунд, после того как через этот порт был послан запрос;
- разрешение приема отдельных типов сообщений ICMP. По умолчанию блокируются все типы сообщений;
- открытие портов — при использовании Standard Profile экран статически открывает порты, которые указаны в списке исключений. Рекомендуется вместо открытия портов включать в список исключения программы, что позволит экрану динамически открывать порты, когда это потребуется соответствующей программе. Если же необходимо открыть порты, то задается номер порта, протокол (TCP или UDP), диапазон или значения IP-адресов (от которых ожидается трафик) и режим (разрешено, запрещено).

Кроме того, можно установить каналы удаленного администрирования и ведения журналов безопасности. Формат файла данного журнала соответствует спецификации Extended Log File Format (расширенный формат файла журнала), предложенной организацией W3C (http://www.w3.org/TR/WD-logfile.html).

Появление персонального МЭ для популярной операционной системы является существенным шагом вперед по обеспечению безопасности индивидуальных пользователей, подключенных к Интернету или другим сетям.

В настоящее время данный персональный МЭ осуществляет ограниченную защиту только от входящего трафика.

Tiny Firewall 2005. Персональный межсетевой экран Tiny Firewall 2005 разработан компанией Tiny Software Inc. (http://www.tinysoftware.com), образованной в 1999 г. Особенностью данного продукта является то, что он изначально создавался для платформы Windows. В настоящее время выпущена уже 6-я версия продукта.

Пользователь с помощью удобного интерфейса может пометить все приложения или группы приложений, заносимые в базу защищаемых приложений.

Этим приложениям или группам приложений ставятся в соответствие определенные правила. Если механизм безопасности экрана находит, что какая-то деятельность соответствует имеющемуся правилу, то он проверяет, находится ли соответствующее приложение в базе (если находится, то к нему применяется соответствующее правило).

Если приложения в базе нет, то механизм прерывает работу и запрашивает пользователя о желаемом действии: закрыть приложение, запустить (включая режим инсталляции) или включить его в базу.

Данный МЭ использует технологию контроля состояний соединений TCP и контроль состояния сеансов UDP и ICMP. Кроме того, используется фильтрация по содержимому. Межсетевой экран обеспечивает как защиту сетевой активности, так и защиту системы, на которой он устанавливается.

Для защиты сетевой активности Tiny Firewall 2005 содержит сетевой механизм безопасности, выполняющий следующие основные функции:

- фильтрация пакетов;
- защита от попыток передачи данных вовне неизвестными процессами;
- защита от использования неизвестными процессами известных приложений;
- избранная фильтрация сетевого трафика на интерфейсах;
- избранная фильтрация трафика, приходящего на интерфейс от множества IP-адресов;
- разделение интерфейсов на зоны;
- временные ограничения на прохождение трафика и т. д.

Защита системы обеспечивается специальными механизмами, которые позволяют установить список доверенных приложений и их прав доступа к системе. Это позволяет изолировать приложения и обеспечить следующие защитные функции:

- предупреждение вставки кода;
- контроль прохождения процессов;
- защиту файлов;
- защиту регистра;
- контроль инсталляции системных служб;
- защиту устройств (предохраняя незаконное использование USB-устройств, COM-портов, модемов и др.);

• полный контроль загрузки DLL и т. д.

Данный персональный межсетевой экран обеспечивает двусторонний контроль трафика, что выгодно отличает его от экрана Windows XP.

Достоинством межсетевого экрана Tiny является и то, что (начиная с версии 5.0) он содержит функции обнаружения и предупреждения вторжений. Обнаружение вторжений реализовано с использованием сигнатурного подхода, для чего используются правила Snort, и при обнаружении реализуется прерывание зловредного трафика. Поддерживается импорт базы данных сигнатур средством SnortImp. Это позволяет защититься от известных видов spyware и дополнять базу своими сигнатурами.

4.1.8. Распределенные межсетевые экраны

Поскольку за последние 5—10 лет МЭ стали одним из основных средств сетевой защиты, вполне естественно, что атакующие стали разрабатывать методы атак «через МЭ» или методы атак самого МЭ как первого рубежа обороны корпоративной сети. Поэтому усилия исследователей были направлены на разработку методов защиты самих МЭ от атак извне. Свидетельством этого является включение в программу исследования качеств МЭ (ежегодно проводимых в США) вопросов устойчивости МЭ к атакам — так называемые тесты на проникновение.

В самой технологии распределенных МЭ можно выделить два основных направления:

• построение системы распределенных межсетевых экранов, реализующих сложную ПБ организации;

• построение системы распределенных компонент МЭ, которые реализуют заданную ПБ.

При построении распределенных систем МЭ их функциональные компоненты распределяются по узлам сети и могут обладать различной функциональностью. При обнаружении подозрительных на атаку признаков управляющие модули распределенного МЭ могут адаптивно изменять конфигурацию, состав и расположение компонент.

В настоящее время распределенные системы МЭ в основном представлены только исследовательскими прототипами.

4.1.9. Межсетевые экраны Web-приложений

Последнее десятилетие характеризуется значительным ростом Web-сайтов. Большинство коммерческих компаний и государственных организаций используют Web-технологии, в числе которых

интернет-банкинг, интернет-магазины, порталы предоставления государственных услуг населению, интернет-сервисы по оплате услуг ЖКХ, покупке билетов через Интернет и многие другие. Количество информационных систем, построенных на Web-технологиях, растет в геометрической прогрессии. Web-приложения могут принимать различные формы, такие, например, как информационный Web-сайт, сайт электронной коммерции, экстранет, интранет, поисковый сайт и т. п. Но, к сожалению, разработчики Web-приложений, стремясь быстрее запустить их в эксплуатацию, зачастую уделяют недостаточное внимание механизмам обеспечения безопасности приложений. Кроме того, современные тенденции развития Web-сервисов показывают отсутствие единых стандартов безопасного программирования Web-приложений, что приводит к ошибкам в разработке ПО и появлению серьезных уязвимостей в Web-сервисах, использующих эти приложения. Положение осложняется тем, что уязвимое Web-приложение может быть легко скомпрометировано без использования специализированных средств, только с помощью браузера. Поэтому количество уязвимостей Web-приложений постоянно растет и, соответственно, растет количество атак на эти приложения. Злоумышленники, пользуясь наличием открытых портов Web-сервера, нацеливают свои атаки именно на приложения.

Необходимость защиты Web-приложений обусловлена следующими основными обстоятельствами:

- корпоративные Web-сайты и Web-приложения должны быть доступны пользователям 24 ч в сутки;
- обычные МЭ не обеспечивают достаточную защиту Web-приложений, так как доступ к сайту извне всегда открыт;
- Web-приложения часто имеют прямой доступ к корпоративным данным и базам данных;
- организации зачастую используют свои приложения, которые недостаточно защищены.

Эти особенности функционирования Web-приложений привели к появлению множества атак прикладного уровня, а именно: *Cross-Site Scripting, SQL Injection, Command Injection, Cookie/Session Poisoning, Parameter/Form Tampering, Directory Traversal/Forceful Browsing, Cookie Snooping, Authentication Hijacking, Log Tampering, Error Message Interception* и др. Поэтому для защиты Web-приложений стали разрабатываться и применяться межсетевые экраны Web-приложений МЭВП (Web Application Firewall, WAF) — новая ветвь технологии защиты, предназначенная для защиты Web-сайтов от атак, причем без модификации прикладного ПО. Межсетевой экран Web-приложений является отдельным устройством, расположенным между клиентом и Web-сервером, которое анализирует данные прикладного уровня для защиты от нарушений ПБ и атак на Web-службы. Такие экраны часто называют МЭ глубокой инспекции пакетов, поскольку они контролируют каждый поступающий на Web-сервер за-

прос и его ответы на уровнях Web-служб (HTTP, HTTPS — Hypertext Transfer Protocol Secure, SOAP — Simple Object Access Protocol, XML-RPC — Extensible Markup Language Remote Procedure Call).

Технология МЭВП базируется на технологии МЭ прикладного уровня. Появление большого числа атак на Web-серверы привело к необходимости перехода к защите Web-приложений. Первые образцы МЭВП появились в 1999 г. Главной их задачей являлась защита от атак, применяющих HTML и XML. Перечислим основные функции, реализуемые МЭВП: разделение сети на отдельные защищенные сегменты, контроль сетевого доступа, фильтрация входящих и исходящих пакетов, обнаружение и предотвращение сетевых атак, фильтрация Web-трафика.

Для координации деятельности защиты Web-приложений был создан консорциум безопасности Web-приложений (Web Application Security Consortium, WASC), который разработал стандарты безопасности Web-приложений.

В 2003 г. был запущен проект Open Web Application Security Project (www.owasp.org), который разработал критерии оценки МЭВП (Web Application Firewall Evaluation Criteria, WAFEC), приведенные на сайте (http://projects.webappsec.org).

Межсетевые экраны Web-приложений располагаются перед Web-сервером (или на нем) и контролируют весь входящий и исходящий трафик. Например, открытая разработка МЭВП ModSecurity (www.modsecurity.org) перехватывает запросы к Web-серверу, на котором он установлен. Клиентский HTTP-запрос сначала обрабатывается модулем ModSecurity, который сравнивает запрос на соответствие своим правилам и, если запрос не блокируется, передает его на обработку Web-серверу. Этот модуль контролирует и ответы Web-сервера (после формировании страницы ответ обрабатывается модулем и, в соответствии с правилами, блокирует или разрешает прохождение ответа). Это можно использовать для замены страниц с ошибками Web-приложений на стандартные страницы (например, страницы с ошибкой 404). Такая фильтрация позволяет уменьшить количество дополнительной информации о Web-приложении, передаваемой клиенту при возникновении ошибок, и затрудняет работу потенциального злоумышленника.

Реализации МЭВП различаются подходами к обнаружению. Одни МЭ просматривают содержимое обмена в поисках сигнатур атак, другие определяют аномальное поведение (отличия от нормального трафика Web-служб). Например, один из первых представителей МЭВП на рынке SecureSphere Web Application Firewall компании Imperva (*www.imperva.com*) содержит функцию динамического профилирования, которая строит модель легитимного поведения приложений и адаптирует эту модель при изменениях в работе приложений. Эта функция является основой многоуровневой архитектуры безопасности, включающей в себя сеть, сервер и приложение.

Запрос пользователя поступает на модуль МЭВП, где анализируется заданная команда (GET, PUT, cookie, POST). Если данные удовлетворяют установленным политикой безопасности правилам, то они передаются в модуль определения поведения (Neural Learning Protocol, NLP), сравнивающий пришедший запрос с данными существующей модели нормального трафика. Модуль либо допускает запрос, либо отвергает его (генерируя сигнал тревоги). Допущенный запрос поступает к Web-серверу.

Модуль BinarySEC для Apache идентифицирует и защищает от зловредного трафика, включающего такие атаки, как SQL and command injections, Cross-Site Scripting, PHP includes, directory traversals, parameter tampering и др. Этот модуль содержит механизм, который обучался поведению приложений, пока не был переведен из состояния обучения в состояние анализа. Такая фаза обучения осуществляется прозрачно для пользователей и администратора Web-сервера.

Во время *фазы обучения* (продолжается несколько дней) этот модуль только регистрирует запросы и генерирует тревоги, которые высвечиваются на консоли и могут быть помечены как нормальный или зловредный трафик. Во время *фазы анализа* модуль работает в режиме блокирования аномального трафика.

Кроме того, современные МЭВП обычно содержат и модуль корреляционного анализа, позволяющего обнаруживать отклонения в поведении приложений, используя множество различных параметров.

Устройства МЭВП, как и большинство других аналогичных устройств, может использовать позитивную или негативную модель безопасности. Используя негативную модель безопасности, оно пропускает по умолчанию все транзакции, кроме тех, которые совпадают с одной или несколькими предварительно определенными сигнатурами атак. Для этого такие МЭ обычно содержат пополняемые базы данных сигнатур. Большинство реализующих позитивный подход устройств содержит механизм обучения, автоматически создающий список достоверных входных данных. Обучение устройства проводится в процессе анализа существующего трафика и создания статистической модели поведения приложений. Построенную таким образом модель впоследствии можно уточнять, добавляя в нее необходимые исключения из построенных правил.

Межсетевые экраны Web-приложений могут быть реализованы в виде как программного средства, так и аппаратно-программного средства.

Рассмотренные технологии, используемые для построения МЭ, обычно обобщаются в виде следующих трех поколений:

1. МЭ фильтрации пакетов;
2. МЭ прикладного уровня;
3. МЭ инспекции состояний.

4.1.10. Новое поколение межсетевых экранов

Широкое применение Web-технологий потребовало пересмотра традиционного подхода к защите сети с помощью МЭ. Пользователям требуется возможность работы в любой точке — в офисе, дома, номере гостиницы или кафе. Кроме того, современные приложения легко обходят традиционные МЭ путем динамической смены портов, использования протоколов SSL и SSH, туннелирования своего трафика через порт 80, применения порта 80 для других протоколов или использования нестандартных портов. Существует множество приложений, которые используют порт 80, что делает возможным прохождение приложений, таких как совместное использование файлов в одноранговых (peer-to-peer) сетях, IM- или IP-телефонии, из Интернета во внутреннюю сеть. Другой проблемой является то, что HTTP-трафик защищается SSL (HTTPS). Протокол HTTPS обычно связан с портом 443/tcp. Так как вся нагрузка этих пакетов зашифрована SSL, то МЭ не могут проводить инспекцию пакетов для определения злонамеренного трафика.

Кроме того, появилось множество интернет-приложений, которые используют для своей работы порты 80 и 443, а также динамические порты. Некоторые из этих приложений необходимы для бизнеса, другие принадлежат пользователям и могут нести угрозу организации.

Межсетевые экраны Web-приложений не могут различить такие приложения, поэтому не обеспечивают необходимый контроль безопасности инфраструктуры организации.

Для эффективного противодействия современным угрозам разработаны МЭ нового поколения (первый промышленный экземпляр — компанией Palo Alto Networks, www.paloaltonetworks.com), обеспечивающие принципиально новый подход, а именно:

- идентификация приложений, а не портов — для идентификации приложения МЭ должен проводить глубокую инспекцию пакетов, а не только использовать пять их основных параметров. Поскольку не существует стандартного пути идентификации приложения, необходимо иметь базу данных сигнатур приложений, включающую в себя идентификаторы для всех общеизвестных и существующих интернет-приложений. Такая база данных должна быть легко расширяема для включения сигнатур новых приложений и приложений пользователей. Идентификация приложений должна игнорировать использование порта 80 и позволять контролировать приложения с изменяемыми номерами портов;
- расширенная инспекция состояний — необходимо контролировать сеансы приложений после того, когда выбраны динамические порты. Межсетевой экран должен иметь возможность обнаруживать аномалии уровня приложений, свидетельствующие о вторжениях или нарушениях политики безопасности;

• расшифрование и зашифрование SSL — должна быть обеспечена способность расшифровать зашифрованную SSL-нагрузку для поиска сигнатур приложения. После проведения инспекции и применения правил политики безопасности разрешенный трафик должен быть снова зашифрован перед посылкой к месту назначения. Функции SSL-прокси, вместе с идентификацией приложений, не должны принимать во внимание использование порта 443;

• идентификация корпоративных пользователей, а не IP-адресов — это позволяет строить и применять политики безопасности, а также формировать отчеты для конкретных пользователей и групп пользователей (служб каталогов);

• сканирование в реальном времени — защита от сетевых атак и вредоносного ПО осуществляется на высоких скоростях и с минимальными задержками;

• высокая производительность (до десятков Гбит/с) для всех функций безопасности — чтобы функционировать в качестве внутреннего экрана в локальной сети или в качестве внешнего экрана на каналах с большой пропускной способностью. Идентификация приложений и обработка SSL на таких скоростях требует, чтобы архитектура экрана базировалась на специальной аппаратуре, а не на стандартных процессорах общего применения;

• простота управления политиками безопасности — традиционные МЭ применяют простейшую модель «разрешить» или «запретить». В этой модели каждый может получить доступ к приложению, который кажется «хорошим». И наоборот, никто не может получить доступ, если приложение кажется «плохим». Эта модель хорошо работает, когда приложения монолитны и не существует множества приложений. Сегодняшняя реальность состоит в том, что приложения могут быть «плохими» для одной организации и «хорошими» для другой. Более того, приложение может быть «хорошим» для одного подразделения организации и «плохим» для других. Поэтому необходима новая модель, которая позволит организациям устанавливать гранулированный уровень контроля для аспектов приложений, которые разрешены соответствующим пользователям.

В качестве примера реализации этих требований кратко рассмотрим основные механизмы межсетевого экрана нового поколения (МЭНП) компании Palo Alto Networks: App-IDTM, User-ID и Content-ID.

App-IDTM — первый механизм классификации трафика, предусматривающий четыре метода для точной идентификации приложения (независимо от порта), протокола, SSL-шифрования или применяемой тактики обхода. Определение приложения является первой задачей и используется для установления соответствия политике безопасности.

User-ID — механизм интеграции со службами директорий организации (такими как Active Directory, eDirectory, LDAP и Citrix). Это

позволяет администратору просматривать и контролировать использование приложений применительно к отдельному пользователю или группе пользователей (а не только по IP-адресу). Информация о пользователе распространяется на все свойства, включая приложения, угрозы, создание политики, расследование следов и формирование отчетов. Механизм User-ID интегрирован с приложениями Active Directory. Это дает возможность определять политику применительно к пользователю или группе пользователей.

Content-ID — потоковой механизм сканирования, использующий общий формат сигнатур для блокирования широкого спектра угроз и ограничения передачи неавторизованных файлов и чувствительных (важных для организации) данных. Кроме того, используется база данных адресов для контроля просмотра Web-сайтов. Данный механизм объединяет три ключевых элемента: конфиденциальность данных (Data Loss Prevention, DLP), предупреждение угроз и URL-фильтрацию.

Защита от угроз реализуется за один проход (просмотр) и объединяется с визуальным представлением приложений и их соответствия политикам безопасности, которые предоставляются App-ID. Фильтрация «за один проход» реализована специально разработанной платформой, обеспечивающей производительность до 5 Гбит/с при применении шифрования. Эта платформа предназначена для обработки сетевого взаимодействия, реализации функций обеспечения безопасности, предупреждения угроз и управления функционированием МЭ. Схема прохождения анализа данных приведена на рис. 4.11.

Данный МЭНП использует следующие механизмы обнаружения и предупреждения атак: анализ декодированного протокола, поиск шаблонов состояния, обнаружение аномальностей используемого

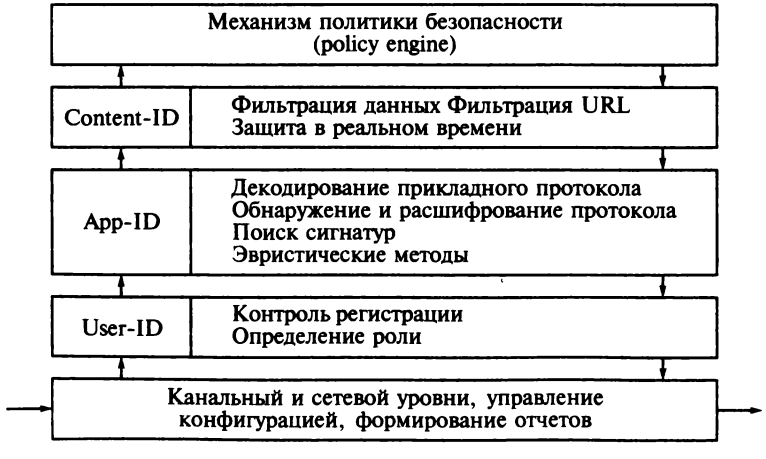


Рис. 4.11. Схема функционирования МЭ нового поколения

протокола, дефрагментация и TCP-реассемблирование, применение базы данных сигнатур и пользовательских сигнатур уязвимостей, блокирование неправильно или зловредно сформированных пакетов, статистическое обнаружение аномалий и эвристический анализ. Защита от эксплойтов, переполнения буфера, атак «отказ в обслуживании» и сканирования портов обеспечивается дополнительными механизмами: декодирование протокола и применение сигнатур, защита от аномальных протоколов (не соответствующих RFC, использующих сверхдлинные URI или длительные входы в FTP), проверка шаблонов состояний для обнаружения атак, охватывающая несколько пакетов (контролируя последовательность прихода пакетов и их последовательные номера).

Общая база данных для фильтрации URL содержит около 20 млн адресов, разделенных на 76 категорий, что позволяет администратору применять гранулированную политику разрешения просмотра Web-сайтов. Кроме того, администратор может сформировать свои категории и добавить их в БД (может быть отдельно сформирована динамическая БД адресов (до 1 млн), сгенерированная из общей БД).

В настоящее время технология App-ID распознает более 750 приложений. Компания добавляет 3 — 5 новых приложений каждую неделю.

4.2. Обход межсетевых экранов

Поскольку МЭ являются первым элементом защиты, то на них, как правило, сосредоточиваются основные усилия атакующих при проведении атак на корпоративные сети. Из анализа существующих тенденций и средств обхода МЭ можно выделить два основных подхода: постепенный подход (firewalking) и туннелирование.

4.2.1. Постепенный подход

Под постепенным подходом (firewalking) понимается методика сбора информации об удаленной сети, защищенной МЭ. Этот метод использует посылку и анализ IP-пакетов подобно утилите traceroute для определения возможности определенного пакета пройти на хост назначения через устройство фильтрации пакетов. Метод позволяет определить открытые порты данного устройства, возможность прохождения пакетов для различных служб через данный порт и т. д.

Утилита traceroute предназначена для перечисления всех хостов на маршруте к цели. Она посылает на хост назначения UDP или ICMP echo (ping) пакеты, в которых постепенно увеличивается значения поля TTL (Time To Live) после каждого удачного шага (по умолчанию

шаг состоит из посылки трех пакетов). Если используется пакет UDP, то номер порта назначения увеличивается на единицу с каждой пробой.

При получении пакета с TTL = 0 маршрутизатор удаляет пакет и посылает на хост-инициатор ICMP error message (time to live exceeded in transit). Это позволят хосту-инициатору узнать, на каком маршрутизаторе пакет удален.

Чтобы удостовериться, что получен ответ от хоста-назначения (ICMP port unreachable или ICMP echo reply), утилита traceroute посылает следующий пакет UDP на порт с большим номером, который вероятнее всего не используется приложениями.

На основе утилиты traceroute разработаны средства, позволяющие проводить требуемый анализ. Одним из таких средств является firewalk. Средство firewalk посылает TCP- и UDP-пакеты с TTL, значение которого на единицу больше целевого шлюза. Если шлюз пропускает трафик, то пакет проходит на следующее устройство, на котором значение TTL будет равно 0. Если же шлюз не пропускает данный пакет, то удаляет его, не посылая никаких сообщений. Подход позволяет определить элементы списка управления доступом шлюза.

4.2.2. Туннелирование

Под термином «туннелирование» понимается общая технология передачи протокола через общую сеть с использованием другого протокола. Предположим, что МЭ разрешает пакеты протокола HTTP. Тогда, если за МЭ во внутренней сети имеется машина, на которой установлен клиент туннелирования, можно организовать обмен данными, которые будут пропускаться МЭ. Такой канал является скрытым. Под скрытым каналом принято понимать любой коммуникационный канал, который может быть использован в процессе передачи информации в обход политики безопасности системы.

В 1996 г. в журнале Phrack magazine (http://www.phrack.org/) было описано средство LOKI, использовавшее ICMP-пакеты для прохождения через межсетевой экран. В 1998 г. появилось средство Rwwwshell, которое туннелировало shell команды внутри HTTP-запросов, причем интерактивный shell использовал возвращаемое сообщение об ошибке 404 для передачи управляющих команд. В дальнейшем эта идея была развита и реализована в средстве httptunnel, которое предназначено для создания HTTP-туннеля типа точка-точка, но может быть использовано и для связывания приложений, работающих на других протоколах (например, SSH или Telnet), для обеспечения неконтролируемого доступа через МЭ.

Таким же средством является libTunnel, использующее HTTP в качестве транспорта для передачи через межсетевой экран разли-

чных сообщений между приложениями. Это программное средство включает в себя библиотеку, которая может быть использована для создания практически любых приложений, использующих HTTP-туннелирование. Алгоритм взаимодействия клиента с сервером при использовании libTunnel включает в себя следующие шаги:

1) если у клиента есть сообщение, то он отправляет HTTP-, POST-запрос;

2) если у клиента нет информации, то он отправляет HTTP-GET-запрос;

3) если у сервера есть сообщение для клиента, то он возвращает его в теле HTTP-ответа.

Эта библиотека позволяет также настроить клиентское приложение для работы с proxy-сервером, чтобы обеспечить корректную работу туннеля через любое количество шлюзов, а серверное приложение для работы с несколькими клиентами одновременно.

Применение туннелирования означает использование инкапсуляции протоколов при межсетевом взаимодействии. В процессе инкапсуляции применяются три типа протоколов: несущий протокол, протокол-пассажир, протокол инкапсуляции.

Транспортный протокол объединяемых сетей является протоколом-пассажиром, а протокол транзитной сети — несущим протоколом.

Пакеты протокола-пассажира помещаются в поле данных несущего протокола с помощью специального протокола инкапсуляции. Пакеты протокола-пассажира не обрабатываются при транспортировке их по транзитной сети. Инкапсуляцию и извлечение пакетов протокола-пассажира выполняют пограничные устройства, располагающиеся на границе между исходной и транзитной сетями, которые указывают в несущих пакетах свои адреса, а не адреса узлов назначения.

По аналогии с коммутацией сетей для использования туннеля сквозь систему защиты необходимо решить следующие задачи:

- разработать несущий протокол, протокол-пассажир и протокол инкапсуляции;
- проанализировать методы обнаружения туннеля и снизить влияние демаскирующих факторов.

Несущий протокол. Очень часто в качестве несущего протокола используется протокол HTTP. На нижних уровнях также существуют механизмы для построения скрытых каналов (например, можно использовать идентификатор IP-пакета, ACK number в TCP-пакетах, сообщения, запросы-ответы DNS), но эти механизмы, как правило, требуют от пользователя привилегированных прав и обладают различными ограничениями. Поэтому для реализации скрытых каналов и средств их конструирования наиболее часто применяется именно протокол HTTP в качестве транспорта для информации. Это объясняется широтой его распространения, легкостью скрытия по-

тока данных, простотой реализации, а также сложностью обнаружения аномалий в протоколе.

Чтобы пакет с данными пересек периметр безопасности, протокол должен быть разрешен для использования субъекту, инициировавшему передачу. В качестве субъекта может выступать пользователь системы или приложение, работающее от его имени (например браузер). Таким образом, чтобы организовать передачу данных по туннелю, надо решить вспомогательные задачи:

- иметь разрешение на отправку данных через периметр безопасности;
- успешно скрывать тот факт, что авторизованный канал используется не по назначению.

Протокол-пассажир оказывает весьма незначительное влияние на общую структуру туннеля, он представляет собой некие данные, которые необходимо передать по туннелю. Это может быть поток данных, создаваемый определенным приложением (например, SSH), или любая другая информация (например, файл), набор управляющих команд и т.д. Вид передаваемой информации существенно влияет на протокол инкапсуляции.

Протокол инкапсуляции. Скрытые каналы можно подразделить на две группы на основании протокола, инкапсулируемого в несущий протокол:

1) канал передачи потока данных приложения (ssh, telnet и т.д.);

2) канал, специально создаваемый злоумышленником.

В первом случае, как правило, создается относительно большой трафик, который демаскирует скрытый канал. Во втором случае обнаружение скрытого канала усложняется, особенно когда канал использует методы стеганографии.

Допустим, агенту заранее предоставлен список макросов:

```
1A cat,
1B echo,
7A «/etc/passwd»,
8A >,
9A «root::0:0:root:/:/bin/sh»
```

Тогда совершенно безобидные запросы вида

```
GET /path/1A-7A.htm HTTP/1.0 и
GET /path/1B-9A-8A-7A.htm HTTP/1.0
```

могут быть пропущены системами защиты.

Данные, передаваемые по скрытому каналу, могут быть обычным текстом, а могут использовать шифрование или методы стеганографии. В качестве контейнера (элемента протокола HTTP, содержащего переданные данные) в запросах HTTP, не содержащих тела сообщения (методы GET, HEAD, DELETE), могут служить:

- строка адреса (URI);
- список заголовков;
- тело ответа, в котором могут содержаться возвращаемые данные (за исключением метода HEAD).

В запросах HTTP, содержащих тело сообщения, к вышеперечисленным параметрам можно добавить тело запроса, что иллюстрируется следующим примером.

Запрос клиента:

```
POST [передаваемые данные]/cgi-bin/server.cgi? [передаваемые данные] HTTP/1.1
Host: <server_host>
Content-Length: <content_length>
Content-Type: application/octet-stream
X-Data: [передаваемые данные]
[передаваемые данные]
```

Ответ сервера:

```
HTTP/1.1 200 OK
Date: Thu, 7 July 2005 06:24:25 GMT
Server: Apache/1.3.27
Content-Length: <content_length>
Content-Type: application/octet-stream
X-Data: [возвращаемые данные]
[возвращаемые данные]
```

Существуют ограничения на размер данных, передаваемых в различных участках HTTP-пакета, которые связаны с существующими реализациями программных продуктов и оговорены в описании протокола.

4.3. Требования и показатели защищенности межсетевых экранов

Государственная техническая комиссия (ГТК) при Президенте Российской Федерации (ныне Федеральная служба по техническому и экспортному контролю, являющаяся преемником ГТК) установила требования и показатели защищенности МЭ в руководящем документе «Средства вычислительной техники. Межсетевые экраны. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации» от 25 июля 1997 г.

Этот документ устанавливает классификацию МЭ по уровню защищенности от несанкционированного доступа к информации на базе перечня показателей защищенности и совокупности описывающих их требований.

Таблица 4.5. **Основные классы защищенности МЭ**

Показатель защищенности	Класс защищенности				
	5	4	3	2	1
Управление доступом (фильтрация данных и трансляция адресов)	+	+	+	+	=
Идентификация и аутентификация			+	=	+
Регистрация		+	+	+	=
Администрирование: идентификация и аутентификация	+	=	+	+	+
Администрирование: регистрация	+	+	+	=	=
Администрирование: простота использования			+	=	+
Целостность	+	=	+	+	+
Восстановление	+	=	=	+	=
Тестирование	+	+	+	+	+
Руководство администратора защиты	+	=	=	=	=
Тестовая документация	+	+	+	+	+
Конструкторская (проектная) документация	+	=	+	=	+

В документе отмечается, что дифференциация подхода к выбору функций защиты в МЭ определяется автоматизированной системой, для защиты которой применяется данный МЭ. Установлено пять классов защищенности МЭ: пятый (низший) класс, применяемый для безопасного взаимодействия АС класса 1Д с внешней средой, четвертый — для АС класса 1Г, третий — для АС класса 1В, второй — для АС класса 1Б и первый (самый высший) — для АС класса 1А. Основные классы защищенности и их показатели приведены в табл. 4.5 (в таблице знак «+» означает наличие или дополнительные требования, знак «=» — требования из низшего класса).

В табл. 4.6 приведены конкретные требования показателя «Управление доступом» для различных классов межсетевых экранов.

При включении межсетевого экрана в автоматизированную систему определенного класса защищенности класс защищенности совокупной автоматизированной системы, полученной из исходной путем добавления в нее межсетевого экрана, не должен понижаться.

Для автоматизированных систем классов 3Б, 2Б должны применяться МЭ не ниже 5-го класса. Для автоматизированных систем классов 3А, 2А (в зависимости от важности обрабатываемой информации) должны применяться МЭ следующих классов:

Таблица 4.6. **Требования показателя «Управление доступом»**

Класс	Требования
5	МЭ должен обеспечивать фильтрацию на сетевом уровне. Решение по фильтрации может приниматься для каждого сетевого пакета независимо на основе, по крайней мере, сетевых адресов отправителя и получателя или на основе других эквивалентных атрибутов
4	Дополнительно МЭ должен обеспечивать: фильтрацию пакетов служебных протоколов, служащих для диагностики и управления работой сетевых устройств, фильтрацию с учетом входного и выходного сетевого интерфейса как средство проверки подлинности сетевых адресов, фильтрацию с учетом любых значимых полей сетевых пакетов
3	Дополнительно МЭ должен обеспечивать: фильтрацию на транспортном уровне запросов на установление виртуальных соединений. При этом учитываются транспортные адреса отправителя и получателя, фильтрацию на прикладном уровне запросов к прикладным сервисам. При этом, по крайней мере, учитываются прикладные адреса отправителя и получателя, фильтрацию с учетом даты/времени
2	Дополнительно МЭ должен обеспечивать: возможность сокрытия субъектов (объектов) и (или) прикладных функций защищенной сети, возможность трансляции сетевых адресов
1	Требования полностью совпадают с требованиями к 2-му классу

- при обработке информации с грифом «секретно» — не ниже 3-го класса;
- при обработке информации с грифом «совершенно секретно» — не ниже 2-го класса;
- при обработке информации с грифом «особой важности» — не ниже 1-го класса.

4.4. Тестирование межсетевых экранов

Поскольку МЭ находится на границе двух сетей и является узловой точкой доступа, то на него возлагается важнейшая задача защиты информации. Организации, использующие МЭ, хотят убедиться в правильности и надежности функционирования основной точки доступа. Особенно это важно в условиях динамических изменений организации. Эта проблема решается при помощи тестирования МЭ.

Для тестирования МЭ используются два основных подхода: тестирование, ориентированное на разработку (назовем его автономное тестирование), и операционное тестирование, называемое тестированием на проникновение (Penetrating Testing).

Цель *автономного тестирования* состоит в исчерпывающей проверке выполнения МЭ своих функций. Для этого осуществляются проверки выполнения МЭ заданной политики безопасности:

1) проверка выполнения требований безопасности:
- выполнение функций контроля доступа;
- контроль соответствия записей в журнал регистрации;
- контроль подсистемы генерации сигналов тревоги;
- контроль доступности;

2) проверка требуемой функциональности:
- проверка выходящих служб (во внешний мир);
- проверка входящих служб (в защищаемую сеть);
- проверка служб, необходимых для взаимодействия с Интернетом.

Кроме того, проверяются правила конфигурирования МЭ и возможности администрирования самого МЭ. Особенно важна проверка производительности МЭ, так как он находится на основной линии обмена данными с внешним миром.

Тестирование выполнения МЭ установленной политики безопасности реализуется записью регистрационных данных (logs) перед МЭ и после него. Для этого могут использоваться следующие атаки и исследования: подделка адресов, перехват сеанса, фрагментация пакетов, маршрутизация источника, DNS-атаки, перенаправление RIP/ICMP, ARP-атаки, сканирование портов, посылка пакетов с нарушениями RFC, насыщение трафика и т. д.

При проведении тестирования, ориентированного на оценку разработки, проводятся следующие действия: сравнение реальной конфигурации МЭ с существующей ПБ, ручное исследование конфигурации через интерфейсы МЭ, операционное тестирование — пробы самого МЭ для поиска открытых служб, проверка правильности предполагаемых действий, исследование правильности контроля разрешенных служб и т. д.

При тестировании производительности определяется скорость выполнения операций межсетевым экраном, не допускает ли МЭ при этом запрещенного трафика. Измеряются и оцениваются задержки, вводимые МЭ при обработке насыщенного трафика.

При тестировании управления проверяются удобство, полнота и эффективность процедур конфигурирования и управления межсетевым экраном.

К требованиям автономного тестирования относятся требования показателя «Тестирование» руководящего документа ГТК, которые приведены в табл. 4.7.

Анализ многочисленных случаев проникновения в защищаемые межсетевыми экранами сети показал, что автономного тестирования

Таблица 4.7. **Требования показателя «тестирование»**

Класс	Требования
5	В МЭ должна обеспечиваться возможность регламентного тестирования: реализации правил фильтрации, процесса идентификации и аутентификации администратора, процесса регистрации действий администратора МЭ, процесса контроля за целостностью программной и информационной частям МЭ, процедуры восстановления
4	Дополнительно должна обеспечиваться возможность регламентного тестирования процесса регистрации
3	Дополнительно должна обеспечиваться возможность регламентного тестирования процесса идентификации и аутентификации запросов
2	Совпадает с требованиями 3-го класса
1	Дополнительно должна обеспечиваться возможность регламентного тестирования процесса централизованного управления компонентами МЭ и графический интерфейс для управления МЭ

МЭ недостаточно для того, чтобы убедиться в правильности их функционирования. Одним из важнейших свойств МЭ должна быть его устойчивость к атакам, направленным на сам МЭ. Это обусловлено тем, что атакующие сначала направляют свои усилия на «обезвреживание» МЭ или его обход. При проведении атаки на МЭ возможны следующие четыре случая:

- МЭ выдерживает атаку и продолжает функционировать;
- атака приводит к краху МЭ, который осуществляет перезапуск и продолжает функционировать;
- атака приводит к краху МЭ, который запрещает прохождение любого трафика;
- атака приводит к краху МЭ, который разрешает прохождение любого трафика.

Каждый из этих случаев должен быть исследован, поскольку оказывает существенное влияние на защищенность сети.

До настоящего момента нет общепринятой методологии *тестирования на проникновение*. Это вызвано как сложностью задачи, так и тем, что специалисты, выполняющие тестирование, чрезвычайно неохотно делятся своими знаниями. Одной из опубликованных методологий тестирования на проникновение является методология, разработанная Мойером (Moyer).

Методология исходит из предпосылки, что попытки атак на МЭ не дают полного представления о защищенности. Тестированию

должна подвергаться вся инфраструктура, защищаемая МЭ. Поэтому предложенная методология включает в себя четыре основные группы действий:

- внешние атаки;
- внутренние атаки;
- совместные атаки (скоординированные атаки снаружи и изнутри);
- анализ разработки и поддерживающих процедур на наличие потенциальных уязвимостей.

Внешние атаки — группа действий, включающая в себя сбор информации (косвенными методами, которые не затрагивают целевую систему и не могут быть ею обнаружены), пробы (сбор информации о целевой системе методами, которые должны быть обнаружены целевой системой), атаки (осуществление попыток вторжений на основе собранных данных и существующих баз данных уязвимостей), нарушение функционирования служб самого МЭ в случае успеха атак.

Внутренние атаки выполняются, исходя из того, что атакующий уже имеет доступ к компьютеру внутренней сети, защищаемой МЭ. При этом могут последовательно использоваться разные уровни имеющегося доступа: пользователя, разработчика ПО или администратора.

В случае совместных атак считается, что внешний атакующий находится в сговоре с пользователем организации, чтобы обмануть систему защиты или передать данные через МЭ. Это становится возможным, если успешными оказались попытки предыдущих групп действий. Обычно на данном этапе используется построение или внедрение системы туннелирования через допускаемый МЭ протокол.

Цель анализа разработки и поддерживающих процедур состоит в поиске методов и способов проникновения, пропущенных при выполнении предыдущих трех групп действий. Для этого анализируется сама разработка МЭ, способы его применения, анализируются административные процедуры и данные, регистрируемые МЭ.

Для проведения тестирования на проникновение используются:

- ручные пробы (проводящий тестирование выполняет команды или запускает свои программы);
- интерактивные действия (проводящий тестирование выполняет действия, анализирует полученные результаты и определяет дальнейшие действия);
- сканеры безопасности различных видов;
- хакерские средства, многие из которых являются общедоступными.

Ручные пробы и интерактивные действия тесно связаны друг с другом и во многих случаях не могут быть разделены. Сканеры безопасности и хакерские средства, как правило, содержат определенную степень автоматизации.

4.5. Отечественный межсетевой экран СППТ-2

В качестве примера отечественной разработки кратко рассмотрим сетевой процессор — МЭ СППТ-2 (www.rtc.ru/production/net-processor.shtml).

Ключевыми особенностями данного МЭ являются:

- режим «стелс» — скрытный режим функционирования;
- многопортовая конфигурация высокоскоростных фильтрующих интерфейсов (до шести портов со скоростью до 1 Гбит/с);
- выделенный управляющий интерфейс;
- масштабируемая производительность.

Основные особенности скрытного режима функционирования МЭ состоят в том, что:

- отсутствуют логические и физические адреса на интерфейсах;
- интерфейсы, подключаемые к защищаемым сегментам сети, работают в режиме приема и обработки всего трафика, передаваемого в данных сегментах;
- в пакете, который прошел обработку и передается на любой из выходных интерфейсов, не изменяются заголовки протоколов и прикладные данные.

Применение скрытного режима функционирования позволяет использовать МЭ в существующих сетях без изменения политики маршрутизации.

Основной функцией МЭ является фильтрация трафика. При этом МЭ разделяет ЛВС на сегменты с различной степенью защищенности и разграничивает доступ к информационным ресурсам сегментов по устанавливаемым администратором правилам. В простейшем случае МЭ разделяет локальную сеть на защищаемый и открытый (подсоединенный к внешним сетям) сегменты. Можно выделить три основных режима функционирования МЭ:

1) режим пакетной фильтрации;

2) режим управления сеансами, фильтрация с учетом состояния виртуального соединения (Statefull Inspection);

3) режим контроля данных прикладного уровня.

Для фильтрации данных в МЭ имеются следующие группы правил:

- MAC-правила — правила фильтрации на уровне кадров Ethernet;
- ARP-правила — правила фильтрации пакетов ARP и RARP;
- IP-правила — правила фильтрации пакетов протокола IPv4. Кроме того, в IP-правилах имеются дополнительные параметры для обработки пакетов протоколов TCP, UDP и ICMP. К этой же группе относятся и так называемые временные IP-правила, действующие на коротком интервале времени для отражения сетевых атак, блокирования абонентов и т. п.
- IPX-правила — правила фильтрации пакетов IPX;
- AP-правила — правила фильтрации прикладного уровня.

При составлении правил используются также специальные структуры (таблицы) «Интервалы времени» и «VLAN-группы», что позволяет привязать правила к определенному временному интервалу и (или) идентификатору VLAN. Каждому правилу ставится в соответствие одно из следующих действий:

- «пропуск» (accept) — передать пакет (внутри МЭ) на выходной фильтрующий интерфейс (интерфейсы) или на следующий уровень фильтрации (для MAC-правил);
- «передача» (pass) — передать пакет (внутри МЭ) на выходной фильтрующий интерфейс (интерфейсы), минуя следующие уровни фильтрации;
- «удаление» (drop) — запретить дальнейшее прохождение пакета.

Основными характеристиками МЭ СППТ-2 являются:

1) многоуровневая скрытная фильтрация пакетов с использованием совокупности критериев;

2) контроль транспортных соединений (до 40 000 TCP-сеансов) путем проверки соответствия каждого пакета контексту соответствующего TCP-сеанса;

3) трансляция сетевых адресов (режим NAT) в режиме «стелс» с выделением «демилитаризованной зоны»;

4) блокировка flood-атак на основе фильтрации аномальной активности сетевых потоков данных;

5) регистрация системных событий и полных заголовков обработанных пакетов на всех уровнях межсетевого взаимодействия;

6) ведение журналов регистрации пакетов и их выгрузка по запросам администратора сети;

7) защита каналов управления на основе алгоритмов шифрования согласно ГОСТ 28147—89 и ГОСТ Р 34.10—2001;

8) выгрузка файлов регистрации событий и пакетов с использованием протоколов FTP и SYSLOG;

9) «зеркалирование» трафика на заданный фильтрующий интерфейс для последующего анализа и проверки;

10) синхронизация системного времени ОС ССПТ-2 по протоколу NTP.

Основная функция МЭ состоит в разделении сегментов автоматизированных систем и фильтрации потоков данных между ними по заданным администратором правилам.

Межсетевой экран обеспечивает фильтрацию на различных уровнях стека протоколов.

На сетевом уровне обеспечивается фильтрация по следующим полям заголовков пакетов протоколов:

- *IP версии 4* — IP-адреса отправителя и получателя, поле флагов TOS, длина IP-пакета, фрагментация пакета (разрешена или запрещена, используется или не используется для данного пакета), время жизни пакета (TTL), протокол верхнего уровня;

• *IP версии 6* — режим «разрешить» или «запретить»;

• *протокол IPX* — адрес сети и (или) узла отправителя, сокет отправителя, адрес сети и (или) узла получателя, сокет получателя, тип пакета;

• *ICMP* — тип сообщения, код сообщения.

На транспортном уровне обеспечивается фильтрация следующих протоколов транспортного уровня, использующих на сетевом уровне протокол IP версии 4:

▪ *TCP* — порт источника, порт назначения, флаги управления потоком (фильтрация по инициатору соединения);

▪ *UDP* — порт источника и порт назначения.

На прикладном уровне обеспечивается фильтрация следующих протоколов прикладного уровня:

▪ *HTTP* (доступ к Web-серверам) — фильтрация по адресам и фрагментам URL, фильтрация по именам и фрагментам имен передаваемых файлов, фильтрация по данным заголовка протокола HTTP;

▪ *SMTP* (электронная почта) — фильтрация по почтовым адресам отправителя (получателя), фильтрация по данным заголовка протокола SMTP;

▪ *FTP* (обмен файлами) — фильтрация по командам протокола GET, PUT; фильтрация по идентификатору и паролю пользователя; фильтрация по именам и фрагментам имен передаваемых файлов; фильтрация по данным заголовка протокола FTP;

• *протоколы передачи SQL*-запросов (SQL*Net, MS-SQL, PostgreSQL, MySQL) — фильтрация по SQL-запросам или их фрагментам.

Кроме того, МЭ СППТ-2 обеспечивает возможность управления сеансами для следующих протоколов:

• *TCP* — контроль неизменности параметров (адреса, порты, интерфейсы) отправителя и получателя на протяжении всего сеанса; контроль корректности переходов между состояниями виртуального TCP-соединения в соответствии с установленными флагами управления; контроль корректности номеров последовательностей;

• *UDP* — контроль неизменности параметров (адреса, порты, интерфейсы) отправителя и получателя на протяжении всего сеанса;

• *ICMP* (только для утилиты *ping*) — контроль неизменности параметров (адрес, идентификатор в заголовке ICMP) отправителя и получателя на протяжении всего сеанса.

Схема проведения фильтрации МЭ СППТ-2 приведена на рис. 4.12.

Исходя из перечисленных характеристик и возможностей, МЭ СППТ-2 может использоваться как:

1) основное средство защиты для реализации различных политик информационной безопасности с помощью:

• фильтрации пакетов на канальном, сетевом, транспортном и прикладном уровнях;

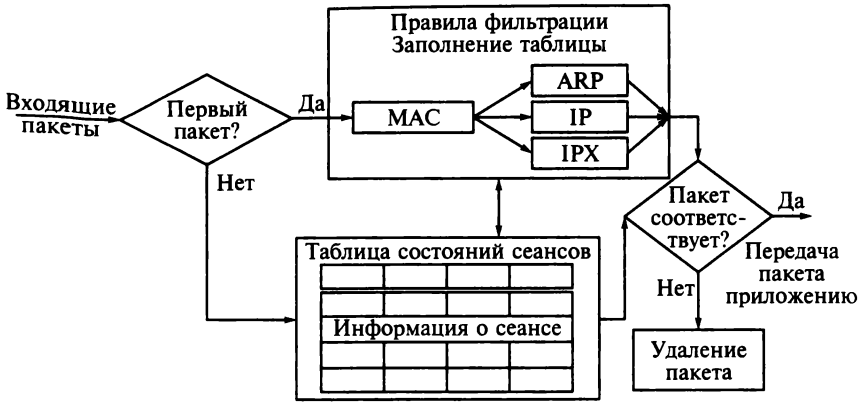


Рис. 4.12. Обработка пакетов в МЭ СППТ-2

- управления транспортными соединениями между отдельными узлами локальной сети или виртуальной локальной сети (VLAN);
- контроля содержимого (контента) данных на прикладном уровне с учетом направления, времени и типа протоколов передачи трафика;

2) дополнительное устройство защиты:

- для обеспечения безопасности функционирования ранее установленных в компьютерной сети средств защиты и устройств маршрутизации;
- мониторинга трафика с возможностью анализа данных регистрации пакетов по различным критериям и интеграции с системой обнаружения вторжений;
- обеспечения функционирования сетевых распределенных приложений и GRID-ресурсов.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Каково основное назначение межсетевых экранов?
2. Могут ли быть применены списки контроля доступа для входящего и исходящего трафиков на одном интерфейсе маршрутизатора?
3. Укажите уровни сетевой модели, на которых применяется фильтрация пакетов.
4. Каковы основные отличия статической и динамической трансляции адресов?
5. Каково значение функции трансляции адресов с точки зрения обеспечения информационной безопасности?
6. Почему порядок правил в списке контроля доступом имеет важное значение?

7. Какой принцип предпочтительнее для фильтрации пакетов: «то, что не запрещено, разрешено» или «то, что не разрешено, запрещено». Почему?
8. Перечислите основные параметры, которые обычно включаются в таблицу состояний.
9. В чем состоят особенности применения службы proxy?
10. Почему существует необходимость применения динамической фильтрации?
11. Сформулируйте понятие виртуального сеанса.
12. Назовите возможные атаки при применении динамического списка контроля доступа.
13. Укажите назначение промежутка времени существования динамического списка контроля доступа.
14. Какие уровни сетевого стека TCP/IP используются при инспекции состояний?
15. Назовите состав основных параметров, которые должны содержаться в таблице состояний.
16. Какое правило «по умолчанию» присутствует в расширенном списке доступа и отсутствует в рефлексивном?
17. Назовите основные отличия между рефлексивными списками контроля доступа и CBAC.
18. Какие технологии фильтрации межсетевых экранов могут быть применены на уровне ядра?
19. Почему основные правила фильтрации персональных межсетевых экранов задаются «по умолчанию»?
20. Назовите основные принципы использование firewalking.
21. Может ли один и тот же протокол быть протоколом-пассажиром и несущим протоколом?
22. Каковы соответствия между классом межсетевого экрана и классом защищаемой им автоматизированной системы?
23. Укажите основные цели тестирования на проникновение.

Глава 5
СИСТЕМЫ ОБНАРУЖЕНИЯ АТАК И ВТОРЖЕНИЙ

— Очень милые стишки, — сказала Алиса задумчиво, — но понять их не так-то легко. (Знаешь, ей даже самой себе не хотелось признаться, что она ничего не поняла.)

— Наводят на всякие мысли — хоть я и не знаю, на какие... Одно ясно: кто-то кого-то здесь убил... А, впрочем, может и нет...

Л. Кэрролл. Алиса в Зазеркалье

Межсетевые экраны, являясь первой линией эшелонированной обороны, не могут обеспечить полную адекватную защиту в силу следующих причин:

- ошибки или недостатки проектирования — различные технологии МЭ не охватывают всех возможностей проникновения в защищаемую сеть;
- недостатки реализации — поскольку каждый МЭ представляет собой сложный программный (программно-аппаратный) комплекс, его реализация содержит ошибки. Кроме того, не существует общей методологии тестирования, которая позволила бы определить качество программной реализации и убедиться, что в МЭ реализованы все специфицируемые свойства;
- недостатки применения (эксплуатации) — применение МЭ для реальной защиты сетей наталкивается на несколько серьезных обстоятельств, к числу которых можно отнести сложность реализации заданной сетевой политики безопасности механизмами МЭ, наличие ошибок конфигурирования МЭ и наличие ошибок администрирования. Администрирование МЭ является достаточно сложной задачей и требует от администратора МЭ определенной квалификации и опыта. Кроме того, атаки могут производиться и со стороны защищаемой сети, что осложняется тем, что они инициируются пользователями, которые, во-первых, имеют доступ к элементам сети, а во-вторых, знают или представляют организацию системы защиты.

МЭ является «единственной точкой», через которую проходят все соединения с внешним миром. Поэтому любая программная ошибка, уязвимость ПО или ошибка конфигурирования МЭ могут привести к проникновению в защищаемую сеть.

Эти и подобные причины вызвали необходимость ведения подробных журналов аудита, в результате анализа которых специалисты (эксперты) делают заключение о совершившихся проникновениях.

Результаты такого анализа используются для устранения причин нарушения или для формирования новых правил (сигнатур), позволяющих защититься от проникновений. Проведение подобного анализа осложняется множеством причин, среди которых основными являются:

- громадный объем данных журналов аудита;
- отсутствие значимых для обнаружения вторжений признаков в журналах;
- сложность анализа данных различных журналов;
- анализ по свершившимся фактам (получение данных об атаке, которая уже удачно прошла, т. е. о вторжении);
- необходимость высокой квалификации проводящего анализ специалиста;
- сложность правил (сигнатур) по обнаруженным признакам атаки.

Поэтому усилия исследователей направлены на разработку процессов обнаружения вторжений и автоматизацию процесса обнаружения.

Подобные системы получили название систем обнаружения вторжений (Intrusion Detection Systems). При рассмотрении систем обнаружения вторжений (СОВ) используется предметная область, элементами которой часто являются общеупотребительные термины.

Поскольку различное толкование терминов приводит к путанице, оговорим отдельные термины, которые будут использоваться в дальнейшем.

Атака — действия, предпринимаемые злоумышленником, против компьютера (или сети) потенциальной жертвы. С точки зрения нейтрального наблюдателя атака может быть безуспешной (неудачной) и успешной (удачной). Успешную атаку называют вторжением.

Вторжение — несанкционированный вход в информационную систему (в результате действий, нарушающих политику безопасности или обходящих систему защиты).

Система обнаружения вторжений — комплекс (аппаратура и программное обеспечение), который по результатам анализа контролируемых и собираемых данных принимает решение о наличии атаки или вторжения.

Ложная тревога (false positive) — генерация сигнала об обнаружении атаки (вторжения), которой не было.

Пропуск (false negative) — пропуск атаки или вторжения (отсутствие сигнала тревоги при наличии вторжения).

Общепринятый термин — система обнаружения вторжений — во многих случаях применяется и для систем обнаружения атак (СОА). В тех случаях, когда не возникает путаница, будем использовать сокращение СОВ.

5.1. Модели систем обнаружения вторжений

История возникновения СОВ насчитывает уже около четверти века. Кратко отметим основные вехи становления технологии обнаружения атак и вторжений (табл. 5.1).

Таблица 5.1. **Хронология развития СОВ**

Год	События
1980	Статья Андерсена (James Anderson) «Computer Security Threat Monitoring and Surveillance». В статье отмечалось, что записи аудита содержат важную информацию, которую можно использовать для выявления злонамеренного поведения пользователей. Впервые появилась концепция обнаружения зловредных и специфических для пользователя событий. Статья дала толчок к государственным заказам на разработку систем обнаружения
1983	Дороти Деннинг (Dorothy Denning) в SRI International начала разработку идеологии обнаружения вторжений. Цель работы — анализ записей аудита государственных компьютерных систем и создание профилей пользователей на основе их текущей активности
1984	Деннинг и Ньюмен (Dorothy Denning and Peter Neumann) разработали первую модель обнаружения вторжений — IDES (Intrusion Detection Expert System), результаты которой были опубликованы в техническом отчете в 1986 г.
1987	Статья Деннинг «An Intrusion-detection model». Первая теоретическая статья с описанием подхода, моделей обнаружения и разработки системы IDES
1998	Появление первых прототипов СОВ
1990-е	Бум разработок коммерческих СОВ, которые спонсировались министерством обороны США
1994	Появление системы NetRanger (Wheelgroup)
1997	Появление системы RealSecure (Internet Security Systems)
2002	Появление систем предотвращения вторжений (систем обнаружения атак с активным противодействием)

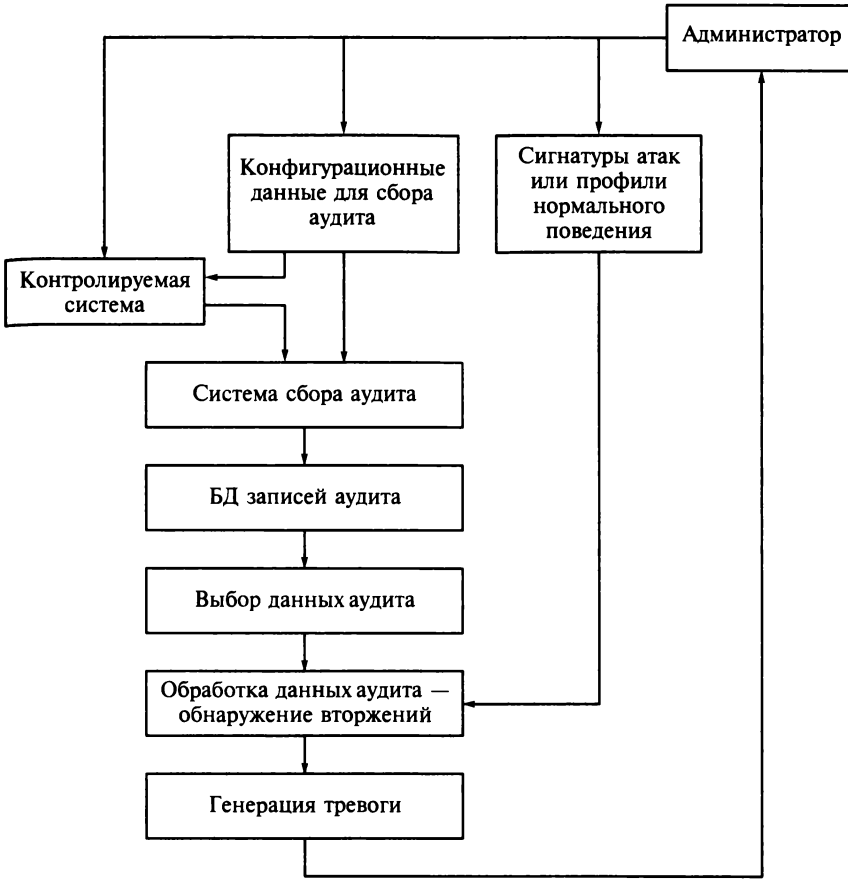


Рис. 5.1. Схема проведения анализа данных аудита

Первые модели и прототипы систем обнаружения вторжений использовали анализ данных аудита компьютерных систем. Схема проведения анализа данных аудита приведена на рис. 5.1.

Рассмотрим основные модели, которые лежат в основе большинства современных СОВ: модель Д. Деннинг и модель, предложенную CIDF (Common Intrusion Detection Framework).

5.1.1. Модель Д. Деннинг

Данная модель базируется на предположении, что нарушения безопасности могут быть обнаружены просмотром записей аудита. Модель включает в себя шесть главных компонентов:

1) субъекты;
2) объекты;
3) записи аудита;
4) профили;
5) записи об аномальностях;
6) правила активности.

Субъекты. В качестве субъектов в модели рассматриваются инициаторы какой-либо активности, обычно это пользователи системы. В качестве субъектов могут выступать процессы пользователей или групп пользователей, а также процессы самой системы.

Объекты. Объектами в модели являются ресурсы (рецепторы действий). Ресурсы включают в себя файлы, программы, сообщения, терминалы, принтеры и структуры, созданные пользователями или их программами.

Записи аудита. Они формируются в следующем формате:

```
<Subject, Action, Object, Exception-Condition,
Resource-Usage, Time-stamp>.
```

Параметры записи аудита: *Subject* — субъект; *Action* — действие, осуществляемое субъектом по отношению к объекту (действиями могут быть: *login*, *logout*, *read*, *execute*, *write* и т.д.); *Object* — объект; *Exception-Condition* — особые условия, возвращаемые системой в ответ на действие (причем эти условия не обязательно возвращаются пользователю); *Resource-Usage* — использование ресурса (промежуток времени, количество записей, количество напечатанных страниц и т.д.); *Time-stamp* — метка даты и времени.

Модель декомпозирует всю деятельность на простейшие действия так, чтобы каждая запись аудита соответствовала одному объекту. Рассмотрим пример, когда пользователь Smith копирует файл task.txt из своего каталога в каталог Lib, куда не имеет доступа по записи. В этом случае должны быть сформированы три записи аудита:

```
Smith, execute, <Lib>task.txt, 0, CPU=0002,
05021121678
Smith, read, <Smith>task.txt, 0, Records=0,
05021121679
Smith, write, <Lib>task.txt, write-violation,
records=0, 05021121680
```

Профили. Профили представляют собой структуры, которые характеризуют поведение субъектов по отношению к объектам в терминах метрик и моделей наблюдаемой деятельности. Наблюдаемое поведение характеризуется терминами статистических моделей и метрик.

В модели предусмотрены три типа метрик:

- счетчик событий — характеризует число записей аудита, удовлетворяющих заданному свойству и возникших за определенный промежуток времени (например, число входов в систему за один час);

• временной интервал — длина интервала времени между двумя связанными событиями;

• измерение ресурса — количество ресурса, потребленного некоторым действием за определенный временной период.

Метрики в модели считаются случайными переменными. В модели рассматривается последовательность из n измерений метрики x: $x_1, x_2, \ldots x_n$. В результате использования соответствующей статистической модели определяется, будет ли новое наблюдение метрики x_{n+1} «ненормально» по отношению к предыдущим.

Рассмотрим модели, которые включены в систему обнаружения.

Функциональная модель, базирующаяся на предположении, что аномальность может быть определена сравнением наблюдаемого значения x с заданными ограничениями. Ограничения определяются из предыдущих наблюдений за переменной того же типа (например, счетчик числа отказов ввода пароля за заданный период).

В качестве модели среднего и стандартного отклонений используются следующие соотношения:

$$S = x_1 + x_2 + \ldots + x_n;$$

$$S^2 = x_1^2 + x_2^2 + \ldots + x_n^2;$$

$$m = s/n;$$

$$d = \text{sqrt}(s^2/n - m^2).$$

Новое значение считается аномальным, если оно выходит из интервала $m + dk$. Из неравенства Чебышева следует, что вероятность того, что значение попадет вне интервала, не превышает $1/k^2$. Например, для $k = 4$, $p = 0{,}0625$.

В модели нескольких переменных вычисляются корреляционные функции для двух и более метрик.

Марковские модели применимы к счетчикам событий, когда каждому типу события (записи аудита) ставится в соответствие переменная состояния и используется матрица переходов состояний. Новое наблюдение считается ненормальным, если его вероятность, определенная через предыдущее состояние и матрицу переходов, мала.

Профиль активности содержит информацию, которая идентифицирует статистическую модель и метрику переменной. Структура профиля содержит семь компонентов, не зависимых от специфики измерения (номера от 1 до 7), и три зависимых компонента (номера от 8 до 10):

1) имя переменной;

2) образец действия — ему сопоставляется 0 или более действий в записях аудита (login, read, execute и т. д.);

3) образец исключения — ему сопоставляется значение поля Exception-Condition в записи аудита;

4) образец использования ресурса — ему сопоставляется значение поля Resource-Usage в записи аудита;

5) период — интервал времени между измерениями (если нет фиксированного временного интервала, то период равен нулю);

6) тип переменной — имя абстрактного типа данных, который определяет частный тип метрики и тип статистической модели (например, счетчик событий со средним значением и дисперсией);

7) порог — определяется для параметра и используется для статистической модели (смысл определяется типом переменной). Для функциональной модели может быть задан верхний или нижний порог, для среднего и дисперсии — k);

8) образец субъекта — сопоставляется поле Subject в записи аудита;

9) образец объекта, которому сопоставляется поле Object в записи аудита;

10) значение — определяются значения текущего или последнего наблюдения и параметры, используемые моделью.

Каждый профиль идентифицируется следующими значениями: имя переменной, образец субъекта и образец объекта.

Модель использует формы шаблонов (образцов), подобных используемым в языке SNOBOL:

`*string*` — строка знаков;

* — любая строка;

— числовая строка;

IN(*list*) — любая строка в списке;

$P \rightarrow name$ — строка, содержащая p, ассоциируется с *name*;

$p1 \vee p2$ — $p1$ или $p2$;

$p1 \wedge p2$ — $p1$ и $p2$;

Not p — не p.

Пример профиля, показывающего измерения количества выходов из терминала пользователя Smith на базе сеансов (с нумерацией компонентов для простоты понимания) приведен в табл. 5.2.

Профили могут быть определены для индивидуальных пар субъект—объект и для агрегированных групп субъектов и объектов. Таким образом, в модели рассматриваются возможные решетки: субъект — объект, субъект — класс объектов, класс субъектов — объект, класс субъектов — класс объектов, субъект (один ко всем объектам), объект (один ко всем субъектам), класс субъектов (ко всем объектам), класс объектов.

Модель предусматривает механизм генерации профилей для новых субъектов и объектов. Для этого могут быть использованы следующие методы:

- ручное создание профиля администратором системы;
- автоматическое создание профиля при появлении поля Create в записи аудита для нового субъекта или нового объекта;
- автоматическое создание профиля по шаблону при первом использовании нового объекта.

Таблица 5.2. **Пример профиля**

№ компонента	Компонент профиля	Значение компонента
1	SessionOutput	Имя переменной
2	`logout`	Образец действия
3	0	Исключение
4	`SessionOutput=` # → Amount	Числовое значение связано с Amount
5		Не используется
6	ResourceByActivity	Тип переменной
7	4	Значение порога
8	`Smith`	Субъект
9	*	Любой объект
10	Record of...	Значение записи

Шаблоны для профилей имеют ту же структуру, что и сами профили, и определяются шаблонами субъектов и объектов.

Для реализации в COB IDES использовались следующие профили:

1) профили активности:

- *LoginFrequency*;
- *LocationFrequency*;
- *LastLogin*;
- *SessionElapsedTime*;
- *SessionOutput*;
- *SessionCPU*, *SessionIO*, *SessionPages*; ...
- *PasswordFails*;

2) профили команд и выполнения программ:

- *ExecutionFrequency*;
- *ProgramCPU*, *ProgramIO*; ...
- *ExecutionDenied*;
- *ProgramResourceExhaustion*;

3) профили активности доступа к файлам:

- *ReadFrequency*, *WriteFrequency*, *CreateFrequency*, *DeleteFrequency*;
- *RecordsRead*, *RecordsWritten*;
- *ReadFails*, *WriteFails*, *DeleteFails*, *CreateFails*;
- *FileresourceExhaustion*.

Записи об аномальностях. Система IDES изменяет профили активности и контролирует аномальность поведения при генерации записи аудита или через заданные интервалы времени. Если обнаружено аномальное поведение, то генерируется запись об аномальности, состоящая из трех компонент:

<Событие, Временная метка, Профиль>.

В этой записи указывается событие, приведшее к аномальности, и профиль активности, по отношению к которому обнаружена аномальность (в действительности IDES в поле «Профиль» заносит указатель на соответствующий профиль из БД).

Правила активности. Они определяют действия, которые должны быть выполнены при генерации записи аудита или записи об аномальности. Правило записывается в виде: *условие* и *тело*. В системе IDES применяются четыре правила.

Правило записи аудита, которое устанавливает соответствие между новой записью аудита и профилем активности (или изменением профиля, записью об аномалии).

Пример.

```
Условие: new audit record
         Audit record matches Profile
         Profile variable-Type = t
Тело:    AuditProcess(audit-Record, Profile); End.
```

Правило периодического обновления параметров активности, которое включается в конце интервала, заданного периодом (или при изменении профиля, контроле аномального поведения).

Пример.

```
Условие: Clock mod p = 0
         Profile.Period = p
         Profile.Variable-type = t
Тело:    PeriodProcesst(Clock, profile); End.
```

Правило записи об аномальности, которое используется при генерации записи об аномальности.

Пример.

```
Условие: new Anomaly-Record
         Anomaly-Record.Profile matches profile-
         pattern
         Anomaly-Record.Event matches event-pattern
Тело:    PrintAlert(1Suspect intrusion of type …',
         Anomaly-Record); End.
```

Правило периодического контроля на аномальность, которое срабатывает в конце заданного временного интервала (или при генерации суммарного отчета за текущий период).

Пример.

```
Условие: Clock mod p = 0
Тело:    Start = Clock - p;
         A = SELECT FROM Anomaly-Records WHERE
         Anomaly-Record.Time-stamp > Start;
         Generate summary report of A.End.
```

Статья с описанием модели Д.Деннинг не только явилась первой теоретической статьей, посвященной СОВ, но и дала толчок развитию исследований в этой области. Подходы, изложенные в статье, и методы данной модели используются в различных СОВ и сейчас.

5.1.2. Модель CIDF

В модели, предложенной CIDF (Common Intrusion Detection Framework, http://www.gidos.org/), выделяются четыре основных компонента:

- генератор событий (e-box, event) — собирает данные для принятия решения анализатором. Данные могут содержать имя контролируемого параметра, его особенности и значения. Сенсор может использовать определенное преобразование данных (например, для преобразования в заданный формат, для уменьшения объема передаваемых данных);
- анализатор (a-box, analyzer) — принимает решение о наличии симптомов атаки на основании данных от сенсоров. Анализатор может выполнять функции преобразования, фильтрации, нормализации и корреляции данных. При обнаружении атаки к данным для генерации тревоги может добавляться семантическое описание обнаруженной атаки. Может быть многоуровневая схема, в которой анализаторы одного уровня передают данные анализаторам более высокого уровня;

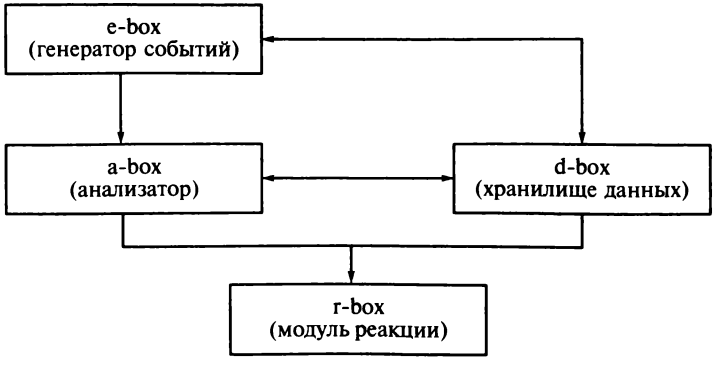


Рис. 5.2. Схема взаимодействия основных компонентов модели CIDF

• хранилище данных (d-box, database) — необходимо для принятия решения и, может быть, данных сенсоров. Кроме того, в хранилище содержатся параметры управления (перечни контролируемых параметров, частота проведения контроля и т. д.) и семантические описания атак. Хранилище может иметь различные формы — от тестового файла до реляционной базы данных;

• модуль реакции системы на обнаруженную атаку (r-box, reaction) — при пассивной реакции система обнаружения вторжений оповещает администратора о начале атаки, используя выдачу сообщения на экран монитора, посылку e-mail, сигнал на пейджер или звонок на сотовый телефон. К активным реакциям относят, например, прекращение процесса, вызвавшего атаку, разрыв сетевого соединения или активную деградацию режима, вызвавшего тревогу.

Схема взаимодействия основных компонентов модели CIDF приведена на рис. 5.2.

5.2. Классификация систем обнаружения вторжений

Для построения таксономии необходимо выбрать критерии, согласно которым будет проводиться классификация. Рассмотрим подход, в котором в качестве таких критериев выбраны типичные функции и особенности проектирования и реализации СОВ (классификация систем обнаружения вторжений представлена на рис. 5.3).

К числу этих функций относятся следующие:

- подход к обнаружению;
- защищаемая система;
- структура СОВ;
- источник данных (для принятия решения);
- время анализа;
- характер реакции.

Подход к обнаружению. Выделяют два основных подхода — обнаружение сигнатур (signature detection, misuse detection), обнаружение аномалий (anomaly detection) — и гибридный подход.

Защищаемая система включает в себя хостовые, сетевые и гибридные СОВ. При контроле на уровне хоста можно рассматривать СОВ следующих подуровней: операционная система, база данных, приложение.

Сетевые СОВ осуществляют сбор и анализ сетевых пакетов, на основании которых проводится обнаружение. Сенсоры таких систем могут быть распределены по элементам сети. Сетевые системы обнаружения вторжений являются системами обнаружения атак.

Хостовые СОВ осуществляют анализ активности отдельного компьютера. В этом случае системе обнаружения вторжений предоставляется гораздо больший набор параметров для анализа. Кроме того, дан-

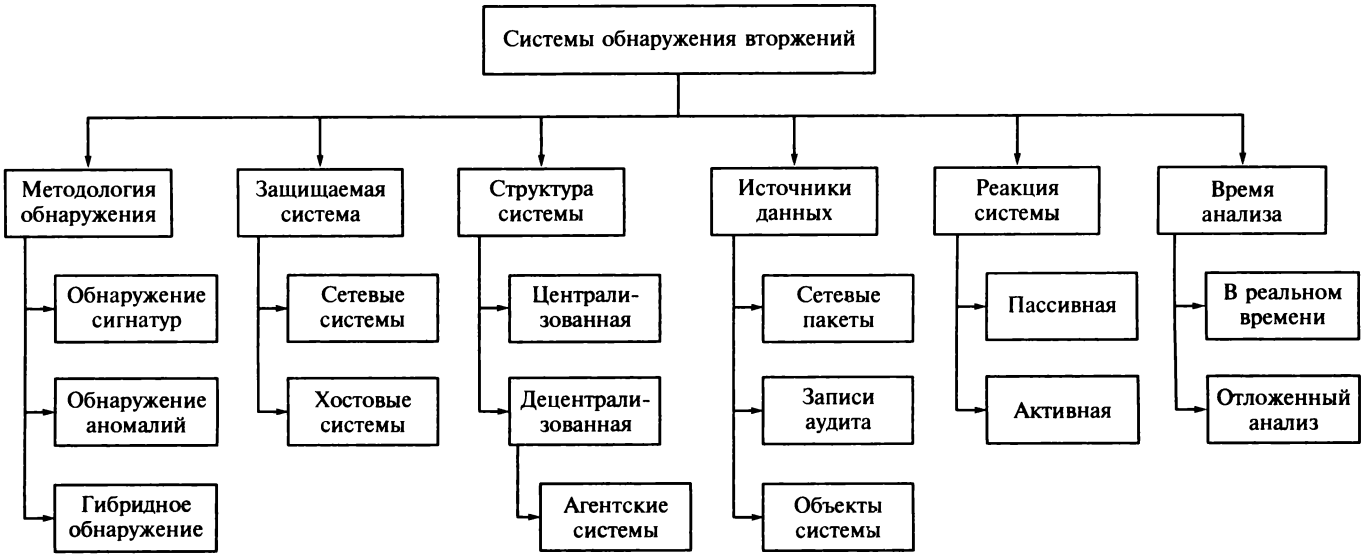


Рис. 5.3. Классификация систем обнаружения вторжений

ные системы обнаружения вторжений могут легко реагировать на обнаруженную атаку. Хостовые СОВ подуровней осуществляют контроль событий, связанных с соответствующим приложением, поэтому хостовые СОВ являются системами обнаружения вторжений. Хостовые СОВ могут проводить и анализ пакетов, прибывающих на данный хост.

Хостовые СОВ, использующие и анализ сетевых пакетов, приходящих на данный хост, являются гибридными, использующими обе технологии одновременно.

Структура. По структуре СОВ подразделяются на централизованные и децентрализованные. Многие из существующих СОВ, как хостовые так и сетевые, имеют монолитную архитектуру, в которой реализуются все операции системы: сбор данных, их анализ и принятие решения, включая генерацию сигнала тревоги. Сложность современных атак, затрагивающих различные аспекты безопасности сети организации, ее сетевых устройств и хостов, а также относительная дороговизна отдельных систем обнаружения вторжений привели к необходимости создания распределенных систем обнаружения. В последние годы появилось значительное число исследований и разработок децентрализованных систем обнаружения вторжений на основе технологии мобильных агентов.

Источник данных. Основными источниками данных для СОВ являются сетевые пакеты и данные аудита. Под записями аудита понимаются не только специально организованные записи собственно аудита, но и системные журналы, которые могут вестись, например, операционной системой. Такие системные журналы могут включать в себя конфигурационные файлы системы, данные для авторизации пользователей и т. д. Эта информация создает основу для последующего принятия решения.

Сенсор интегрируется с компонентом, ответственным за сбор данных — генератором событий (e-box). Поэтому способ сбора информации определяется политикой генерации событий, которая определяет режимы фильтрации просматриваемых событий. Генератор событий (ОС, сеть, приложение) генерирует множество событий, зависящих от политики, которые могут записываться в журналы (системные и аудита) или определяться сетевыми пакетами. Данные в зависимости от политики могут сохраняться как внутри защищаемой системы, так и вне ее. В отдельных случаях данные могут не сохраняться, но тогда потоки данных о событиях передаются прямо в анализатор. Это касается, в частности, сетевых пакетов.

Существуют проблемы, связанные с обработкой следов аудита.

Сохранение отчетов аудита в единственном файле может быть небезопасно, так как нарушитель может использовать его в своих интересах. Лучше иметь определенное число копий, распределенных по сети, но это увеличивает нагрузку на сеть и систему.

С функциональной точки зрения регистрация каждого события означает значительное увеличение нагрузки на систему. Сжатие жур-

налов регистрации также увеличивает системную нагрузку, высвобождая место на дисках.

Определение того, какие события необходимо включать в аудит, достаточно затруднительно, так как определенные типы атак (вторжений) могут остаться необнаруженными.

Трудно предсказать возможный объем файлов аудита и определить периоды их сохранения.

Системы обработки журналов регистрации уязвимы к атакам отказа в обслуживании.

Время анализа. СОВ могут функционировать непрерывно или периодически получая информацию для обработки (real time, interval-based). Это разделение определяет применение двух различных подходов к обнаружению, которые мы будем называть режимом реального времени и отложенным режимом.

При обработке в реальном масштабе времени осуществляется текущая верификация системных событий для хостовых систем и анализ сетевых пакетов для сетевых СОВ. Из-за вычислительной сложности в сетевых СОВ используемые алгоритмы ограничиваются скоростью выполнения и своей эффективностью, поэтому часто используются наиболее простые алгоритмы.

Системы обнаружения атак должны функционировать только в реальном масштабе времени, а системы обнаружения вторжений могут функционировать как в реальном, так и в отложенном режимах.

Характер реакции. По характеру реакции различают пассивные (генерирующие только сигналы тревоги) и активные (выполняющие обнаружение и реализацию реакции на атаки) СОВ. В качестве реакции могут использоваться, например, попытки установить соответствующие «заплатки» в ПО перед взломом или блокировки атакуемых служб. В течение нескольких лет на страницах Интернета даже обсуждалась проблема активных атакующих действий против нарушителя. Пассивные СОВ могут выдавать сигнал тревоги на консоль администратора, выдавать сообщение на пейджер, посылать сообщение по электронной почте и даже выполнять звонок на заданный сотовый телефон.

5.3. Обнаружение сигнатур

В настоящее время нет точного толкования слова «сигнатура». Обычно под сигнатурой понимается набор битовых критериев, используемых в качестве шаблона для обнаружения атак в трафике. Ряд производителей СОВ употребляет термин «фильтр» в качестве логического правила обнаружения в комбинации с предполагаемым действием. Далее будем использовать термин «сигнатура».

Под *сигнатурой* будем понимать множество условий, при удовлетворении которых наступает событие, определяемое как атака

Таблица 5.3 **Достоинства и недостатки подхода совпадения с шаблоном**

Достоинства	Недостатки
Простота задания правил обнаружения Прямая связь с используемой данной атакой уязвимостью Высокая надежность Применим для всех протоколов	Может привести к ложному срабатыванию, если шаблон не уникален Может потребоваться множество шаблонов для одной уязвимости Ограниченность подхода, связанная с анализом только одного пакета Существуют варианты «обхода» данного метода

или вторжение (обнаружение сигнатур обычно связывают только с совпадением с образцом). Для обнаружения сигнатур могут использоваться следующие подходы.

Совпадение с шаблоном — обнаружение в этом случае базируется на поиске фиксированной последовательности байтов в рассматриваемом элементе данных (например, в единичном пакете). Как правило, шаблон сопоставляется только в том случае, если подозреваемый пакет ассоциирован с определенной службой (предназначен определенному порту). Существуют протоколы, которые не имеют определенных портов. Пример условий сигнатуры: пакет IPv4 содержит указание на протокол TCP и порт назначения 2222, а полезная нагрузка содержит строку foo — «генерировать тревогу». Основные достоинства и недостатки данного подхода приведены в табл. 5.3.

Совпадение с шаблоном состояния — по одному пакету устанавливается состояние потока данных. Появление другого пакета (или пакетов), который соответствует данным состояния, считается атакой. Достоинства и недостатки данного подхода перечислены в табл. 5.4.

Анализ на основе шаблона используемого протокола — для формирования состояния используется декодирование различных элементов протокола. При декодировании протокола СОВ применяет правила, определенные RFC для нарушений. В некоторых случаях эти нарушения могут находиться в определенных полях протокола, что требует более детального анализа. Основные достоинства и недостатки данного подхода приведены в табл. 5.5.

Эвристический подход — использует логические правила, полученные эвристически. Например, для обнаружения сканирования портов можно использовать порог затронутых портов целевой системы. Кроме того, сигнатура может быть ограничена заданием только

Таблица 5.4. **Достоинства и недостатки подхода совпадения с шаблоном состояния**

Достоинства	Недостатки
Большая эффективность по сравнению с совпадением с шаблоном Прямая связь с используемой данной атакой уязвимостью Высокая надежность Применим к множеству протоколов Затрудняет «обход» системы анализа	Может привести к ложному срабатыванию, если шаблон не уникален Модификация атаки может привести к пропуску обнаружения Может потребоваться множество шаблонов для одной уязвимости

определенных типов пакетов (например, SYN-пакетов). В табл. 5.6. приведены основные достоинства и недостатки данного подхода.

Возможны различные варианты разработки сигнатур, которые и определяют соответствующие технологии обнаружения. Для их иллюстрации воспользуемся уязвимостью переполнения буфера в Microsoft RPC DCOM (бюллетень MS03-026). Впервые этот бюллетень и соответствующее исправление безопасности были выпущены корпорацией Microsoft 16 июля 2003 г., чтобы устранить уязвимое место в системе безопасности интерфейса DCOM (Distributed Component Object Model) RPC (Remote Procedure Call) системы Windows. Это исправление до сих пор надежно закрывает брешь в системе безопасности. Однако в разделах «Факторы, снижающие опасность» и «Временное решение» первой версии бюллетеня были указаны не все порты, которые могут представлять опасность. Поэтому корпорация Microsoft выпустила обновленную версию бюллетеня,

Таблица 5.5. **Достоинства и недостатки анализа на основе шаблона используемого протокола**

Достоинства	Недостатки
Если протокол хорошо определен, то минимизируются пропуски вторжений Прямая связь с используемой данной атакой уязвимостью Метод может обнаруживать варианты атаки Надежен, если определены правила протокола	Большое число пропусков, если RFC позволяет различные интерпретации Сложность формирования сигнатур

Таблица 5.6. **Достоинства и недостатки эвристического подхода**

Достоинства	Недостатки
Сигнатуры могут отражать сложные взаимосвязи Возможность обнаружения атак, не обнаруживаемых предыдущими методами	Разработка эвристик трудоемка и требует высокой квалификации Эвристические алгоритмы требуют приспособления к соответствующему трафику

в которой указаны все порты, используемые службами RPC, чтобы пользователи, применяющие временные решения проблемы перед установкой исправления, располагали всей необходимой для защиты компьютера информацией. Сетевой протокол DCE/RPC, в частности, определяет вызовы сетевых функций, предоставляемых различными интерфейсами Microsoft, и передачу соответствующих аргументов этим функциям. Протокол RPC используется операционной системой Windows. Он обеспечивает взаимодействие между процессами и позволяет программам, запущенным на одном компьютере, беспрепятственно выполнять код на удаленном компьютере. В его основе лежит другой протокол — OSF (Open Software Foundation) RPC. Используемый в Windows протокол RPC представляет собой расширенную версию последнего.

Одно из таких расширений, которое отвечает за обмен сообщениями по TCP/IP, является уязвимым местом. Обработка некорректно составленных сообщений может привести к сбою в работе системы. Это уязвимое место связано с интерфейсом DCOM, который ведет прослушивание портов RPC. Этот интерфейс обрабатывает запросы на активацию объектов DCOM, отправляемых клиентскими компьютерами серверу (например, запросов путей в стандартном формате записи имен UNC). Злоумышленник, которому удастся воспользоваться этим уязвимым местом, получит возможность запускать на компьютере пользователя собственный код от имени учетной записи «Локальный компьютер». Он сможет выполнять в системе любые действия, в том числе устанавливать программы, просматривать данные, изменять и удалять их, а также создавать новые учетные записи с полными правами. Чтобы воспользоваться уязвимостью, злоумышленник должен направить на нужный порт RPC удаленного компьютера специальный запрос.

Были выявлены слабые места в той части службы удаленного вызова процедур Windows (RPCSS), которая относится к инструкциям RPC по активации DCOM. Злонамеренный пользователь, используя эти слабые места, может запустить код с правами на доступ к локальной системе на пораженном компьютере или вызвать прекращение работы службы RPCSS. Затем этот злонамеренный пользователь может выполнить на компьютере любое действие, включая установку про-

грамм, просмотр, изменение и удаление данных или создание новых учетных записей с полными правами. Чтобы воспользоваться этими уязвимыми местами, злоумышленник должен создать исполняемую программу для отправки неправильных сообщений RPC, которые будут направляться на конечную службу RPCSS уязвимого сервера.

Рассмотрим подробнее схему данной атаки. RPC-сервер предоставляет различные интерфейсы, каждый из которых определяется уникальным идентификатором IID из 16 байт. Имевший рассматриваемую уязвимость интерфейс ISystemActivator имел следующий IID:

```
000001a0-0000-0000-c000-000000000046
```

Каждый интерфейс предоставляет семейство функций, которые определяются значением идентификатора opnum. Для данной уязвимости значение opnum = 4 в вызываемом интерфейсе. Запрашиваемая клиентом комбинация interface/opnum допускает различные множества аргументов для функций. Клиент сначала устанавливает TCP соединение с портом 135/tcp (или портами 139, 445, 593/tcp, 135/udp) сервера. Схема проведения клиентом атаки на Microsoft RPC DCOM показана на рис. 5.4.

После установления соединения клиент посылает BIND для присоединения к желаемому интерфейсу. Затем клиент посылает запрос REQUEST для вызова определенной функции opnum в выбранном интерфейсе. Содержимое запроса включает в себя данные, которые интерпретируются сервером как аргументы вызываемой функции. Одним из таких аргументов является имя файла, которое задается в виде: \\server\filename. Причем Windows ожидает, что имя server не будет превышать 32 байт. Если атакующий посылает длинное имя, то благодаря уязвимости в используемой системной библиотеке возникает переполнение буфера, позволяющее атакующему выполнить свой собственный код, вставленный в это длинное имя.

Рассмотрим различные сигнатуры для обнаружения этой атаки.

Сигнатуры эксплойта опираются на характеристики атаки, которые позволяют однозначно идентифицировать атаку. Для рассматриваемого примера параметрами уязвимости, например, будут:

- запрошенный уязвимый интерфейс ISystemActivator;
- запрошенная уязвимая функция (opnum = 4);
- длинное имя файла;
- наличие в имени файла машинных команд, реализующих уязвимость.

Ключевым параметром, который отсутствует при нормальной работе, является наличие в имени файла машинных команд, реализующих уязвимость. Поэтому сигнатурой уязвимости может служить строка, содержащая машинные команды для реализации атаки. Данная сигнатура будет работать только при использовании известного эксплойта. Если атакующий поменяет машинные команды, то сигнатура не обнаружит модернизированную атаку. Другим способом обхода данной

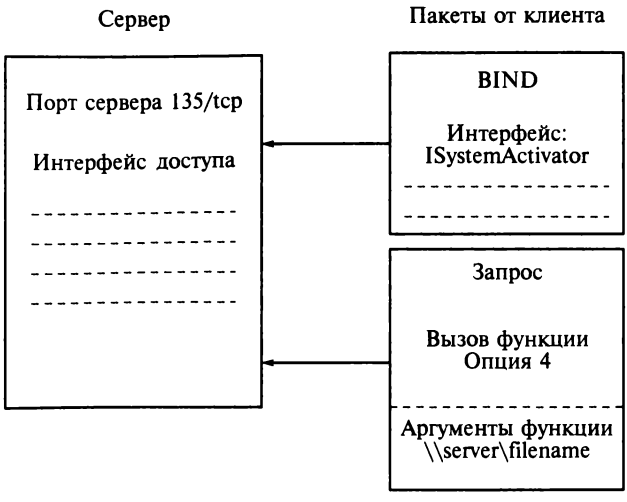


Рис. 5.4. Схема атаки на Microsoft RPC DCOM

сигнатуры является фрагментация пакета, содержащего REQUEST так, чтобы машинный код содержался в разных фрагментах.

Сигнатуры уязвимости опираются на особенности конкретной уязвимости, т.е. на те параметры или действия, которые необходимо выполнить для использования данной уязвимости. Для рассматриваемого примера запишем основные условия, которые необходимы для успеха атаки:

- установление соединения с портом 135/tcp;
- посылка BIND к уязвимому интерфейсу;
- посылка REQUEST с opnum = 4;
- наличие параметров в списке аргументов;
- наличие длинного имени сервера.

В данном случае отличительной характеристикой уязвимости является длинное имя сервера. Поэтому сигнатурой может быть контроль длины имени сервера (не более 32 байт).

Сигнатуры аномалий протоколов иногда трактуются как обнаружение аномалий протокола (Protocol Anomaly Detection, PAD). Для разработки таких сигнатур необходимо провести анализ реализации рассматриваемого протокола на соответствие RFC. Не все протоколы могут быть проверены по RFC, кроме того, некоторые программные продукты допускают трафик, нарушающий спецификации.

Для рассматриваемой уязвимости выработка соответствующей сигнатуры не представляется возможной в силу следующих причин:

- данный протокол не имеет RFC;
- исходные тексты протокола недоступны;
- отсутствует информация о том, как приложение обрабатывает входные данные.

Чтобы проиллюстрировать построение сигнатуры аномалии протоколов, рассмотрим использование URL для атак в протоколе HTTP. Для этого рассмотрим следующие URL:

```
1. http://www.google.com/search?q=%27+having+1%3D1--
2. http://<target>/productcard/pc/Custvb.asp?Email=
%27+having+1%3D1--
```

Данные URL являются похожими (оба не нарушают RFC), но первый запрос является законным, а во втором запросе используется атака SQL Injection.

Сигнатуры политики. Рассмотрим сигнатуру Snort (см. Приложение) для рассматриваемой уязвимости (для удобства чтения отдельные поля расположены на разных строках):

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 135
(msg: "NETBIOS DCERPC ISystemActivator bind
attempt";
flow: to_server, established;
content:"|05|"; distance:0; within:1;
content:"|06|"; distance:1; within:1;
byte_test: 1,&,1,0,relative;
          content:"|A0010000000000C000000000000046
          |"; distance:29; within:16;
reference:cve,CAN-2003-0352; classtype:attempted-
admin; sid:2192;)
```

Видно, что данная сигнатура предназначена для простого поиска заданных строк в определенном месте полезной нагрузки TCP-пакета. Поэтому при законном использовании рассматриваемой функции данного интерфейса будет генерироваться сигнал тревоги. Кроме того, приведенную сигнатуру Snort можно обойти, используя следующие возможности: во-первых, применяя фрагментирование пакета, во-вторых, используя возможность, предоставляемую протоколом, который допускает REQUEST нескольких интерфейсов. Поиск заданных строк сигнатурой Snort на определенных позициях приведет к пропуску атаки.

В рассматриваемом случае сигнатурой политики может быть решение администратора о запрещении соответствующих DCOM-соединений до устранения опубликованной уязвимости разработчиком продукта.

5.4. Система обнаружения вторжений Snort

Сетевая система обнаружений Snort была разработана Мартином Рош (Martin Roesch) как упрощенная СОВ в виде сетевого анализа-

тора (www.snort.org). Благодаря постоянным усовершенствованиям и расширениям Snort все более превращается в полнофункциональную систему анализа IP-трафика и записи пакетов в реальном времени. В настоящее время Snort является одним из примеров совместной работы интернет-сообщества. Поскольку СОВ Snort является открытой разработкой, множество исследователей принимают участие в разработке новых дополнительных модулей и усовершенствовании старых.

По своему замыслу Snort является переносимой системой и работает на многих операционных системах. В настоящее время существует Snort для архитектур x86 Linux, free BSD, NetBSD, Open BSD и Windows.

Система Snort может работать в трех основных режимах: анализатор пакетов, регистратор пакетов и сетевая система обнаружения вторжений. Кратко рассмотрим основные компоненты Snort. Сначала входные данные (сетевые пакеты) обрабатываются декодером пакетов. Если Snort используется только как анализатор пакетов, то декодированные данные форматируются и выводятся на консоль. Если Snort используется как регистратор пакетов, то декодированные данные, в зависимости от того что задано в командной строке, или сохраняются в двоичном формате, или преобразуются в формат ASCII и записываются в файл на диске. Если Snort используется как сетевая СОВ, то обработка продолжается. Декодированные данные передаются препроцессорам, которые указаны в файле конфигурации *snort.conf*. Затем данные передаются процессору обнаружения, который осуществляет их сравнение с параметрами наборов правил, которые заданы в файле конфигурации. В случае совпадения данных происходит пересылка компонентам регистрации и сигнализации (в зависимости от настроек происходит регистрация или генерация сигналов тревоги).

Схема основных модулей и их взаимодействия приведена на рис. 5.5.

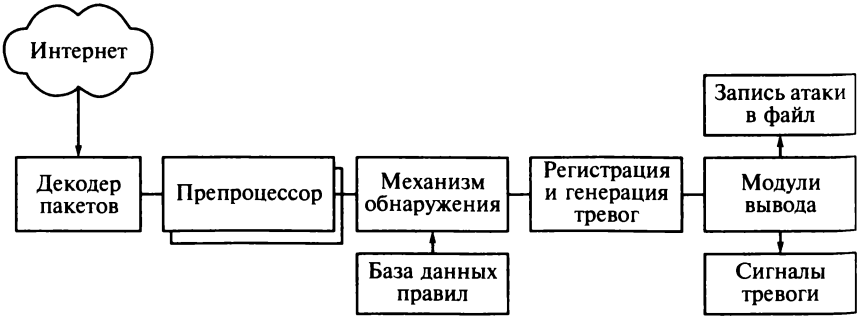


Рис. 5.5. Основные компоненты СОВ Snort

Основным файлом является файл конфигурации, первые строки которого имеют следующий вид:

```
##################################################
# This file contains a sample snort configuration.
##################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your
own custom configuration:
#
# 1) Set the network variables.
# 2) Configure the decoder
# 3) Configure the base detection engine
# 4) Configure dynamic loaded libraries
# 5) Configure preprocessors
# 6) Configure output plugins
# 7) Customize your rule set
# 8) Customize preprocessor and decoder rule set
# 9) Customize shared object rule set
##################################################
```

Как видно из файла конфигурации, для работы Snort необходимо выполнить указанную последовательность действий. Кратко рассмотрим основные модули, которые требуется указать в файле конфигурации.

5.4.1. Декодер пакетов

Декодер пакетов определяет протокол, содержащийся в рассматриваемом пакете, и проверяет соответствие данных этому протоколу. Декодер может генерировать свои собственные сигналы в случае неправильно сформированных заголовков пакетов, слишком больших пакетов, наличия необычных или неправильных опций TCP, установленных в заголовке, и в других аналогичных случаях. Посредством изменения файла конфигурации можно включать или отключать различные наборы сигнатур для этих полей пакетов.

5.4.2. Препроцессоры

Препроцессорами являются подключаемые модули Snort, которые позволяют анализировать входные данные различными способами. Препроцессоры расширяют функциональность Snort, давая возмож-

ность пользователям и программистам легко вводить дополнительные модули. Препроцессоры выполняются перед вызовом механизма обнаружения, но после декодирования пакета. Препроцессоры конфигурируются и загружаются при использовании ключевого слова preprocessor и имеют следующий формат:

```
preprocessor <name>: <options>
```

Система Snort содержит препроцессоры нормализации пакетов, чтобы избежать методов обхода СОВ, которые будут рассмотрены в подразд. 5.7. Текущая версия предусматривает нормализацию следующих протоколов: IPv4, IPv6, ICMP (v4 и v6) и TCP. Модули нормализации пакетов также называются препроцессорами. Пример их конфигурирования:

```
preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6
```

Кроме того, Snort содержит широкий выбор препроцессоров, которые можно использовать в различных режимах работы. Препроцессорами текущей версии 2.9.2 являются: Frag3, Stream5, sfPortscan, RPC Decode, Performance Monitor, HTTP Inspect, SMTP Preprocessor, POP Preprocessor, IMAP Preprocessor, FTP/Telnet Preprocessor, SSH Preprocessor, DNS Preprocessor, SSL/TLS Preprocessor, ARP Spoof Preprocessor, DCE/RPC 2 Preprocessor, Sensitive Data Preprocessor, Normalizer Preprocessor, SIP Preprocessor, Reputation Preprocessor, GTP Decoder and Preprocessor, Modbus Preprocessor и DNP3 Preprocessor.

Все препроцессоры по выполняемым функциям можно условно разбить на три группы:

1) сборка (реассемблирование) пакетов (препроцессоры Frag3, Stream5);

2) нормализация протоколов (препроцессоры RPC decode, HTTP inspect, FTP/Telnet, SSH, SMTP, POP, DNS и т. п.);

3) обнаружение известных аномалий в трафике (препроцессор sfPortscan).

Кратко рассмотрим некоторые из этих препроцессоров.

5.4.3. Препроцессоры сборки пакетов

Препроцессор дефрагментации *Frag3* позволяет осуществлять обработку, ориентированную на целевую систему. Он заменил препроцессор Frag2 и имеет дополнительные свойства: более быстрое выполнение и обработка, ориентированная на целевую систему для защиты от методов обхода и обмана СОВ. Идея учета целевой системы состоит в учете особенностей построения сетевых стеков реаль-

ных хостов защищаемой сети и снабжение СОВ топологической информацией.

Основная идея ориентации на целевую систему состоит в моделировании поведения стека целевой системы, в частности особенностей обработки фрагментированных пакетов. Поэтому СОВ обрабатывает фрагментированные пакеты так же, как и защищаемая система. Кроме того, используется защита от обхода СОВ на основе TTL.

Для задания типа защищаемой системы используется параметр policy <type>. По умолчанию задается тип bsd. Указанные типы политик определяются особенностями различных ОС (их стратегией действий при обработке перекрывающихся фрагментов). Основные платформы типов целевых систем, которые может контролировать Snort, приведены в табл. 5.7.

Таблица 5.7. **Платформы целевых систем, защищаемых Snort**

Целевая платформа	Тип целевой системы для Snort
AIX 2	BSD
AIX 4.3 8.9.3	BSD
Cisco IOS	Last
FreeBSD	BSD
HP JetDirect (printer)	BSD-right
HP-UX B.10.20	BSD
HP-UX 11.00	First
IRIX 4.0.5F	BSD
IRIX 6.2	BSD
IRIX 6.3	BSD
IRIX64 6.4	BSD
Linux 2.2.10	Linux
Linux 2.2.14-5.0	Linux
Linux 2.2.16-3	Linux
Linux 2.2.19-6.2.10smp	Linux
Linux 2.4.7-10	Linux
Linux 2.4.9-31SGI 1.0.2smp	Linux

Целевая платформа	Тип целевой системы для Snort
Linux 2.4 (RedHat 7.1-7.3)	Linux
MacOS (version unknown)	First
NCD Thin Clients BSD	BSD
OpenBSD (version unknown)	Linux
OpenBSD (version unknown)	Linux
OpenVMS 7.1	BSD
OS/2 (version unknown)	BSD
OSF1 V3.0	BSD
OSF1 V3.2	BSD
OSF1 V4.0,5.0,5.1	BSD
SunOS 4.1.4	BSD
SunOS 5.5.1,5.6,5.7,5.8	First
Tru64 Unix V5.0A,V5.1	BSD
Vax/VMS	BSD
Windows (95/98/NT4/W2K/XP)	Windows

Препроцессор содержит два препроцессора: глобальной конфигурации (frag3_global) и обработки (frag3_engine).

Примеры конфигурирования препроцессора:

```
preprocessor frag3_global: max_frags 65536
                prealloc_frags 262144
preprocessor frag3_engine: policy linux
       bind_to { 10.1.1.11/32, 10.1.1.13/32 }
       detect_anomalies
```

Препроцессор позволяет задавать различные типы для отдельных компьютеров защищаемой сети. Например:

```
preprocessor frag3_engine: policy linux, bind_to
192.168.1.0/24
preprocessor frag3_engine: policy first, bind_to
[10.1.47.0/24,172.16.8.0/24]
```

Препроцессор *Stream5* предназначен для обработки пакетов TCP, ориентированной на целевую систему. Он может контролировать сеансы как TCP, так и UDP. Сеансы TCP идентифицируются по установленному соединению. Сеансы UDP определяются как серия

пакетов UDP между двумя абонентами, использующих одно и то же множество портов. Сообщения ICMP допускаются, если они связаны с соответствующими TCP и UDP-сеансами.

Препроцессор обнаруживает такие аномалии TCP-протокола, как наличие данных в пакете с флагом SYN, посылка данных вне окна TCP и др.

```
preprocessor stream5_global: \
[track_tcp <yes|no>], [max_tcp <number>], \
[memcap <number bytes>], \
[track_udp <yes|no>], [max_udp <number>], \
[track_icmp <yes|no>], [max_icmp <number>], \
[track_ip <yes|no>], [max_ip <number>], \
[flush_on_alert], [show_rebuilt_packets], \
[prune_log_max <bytes>], [disabled]
```

Этот препроцессор позволяет конкретизировать политику обработки фрагментированных пакетов на целевой системе посредством параметра policy <policy id>. Возможные варианты политик приведены в табл. 5.8.

Таблица 5.8. **Политики целевых систем, которые защищает Snort**

Имя политики	Операционная система
first	Выбор первого из перекрывающихся фрагментов
last	Выбор последнего из перекрывающихся фрагментов
bsd	FresBSD 4.x и выше, NetBSD 2.x и выше, OpenBSD 3.x и выше
linux	Linux 2.4 и выше
old-linux	Linux 2.2 и более ранние
windows	Windows 2000, Windows XP, Windows 95/98/ME
win2003	Windows 2003 Server
vista	Windows Vista
solaris	Solaris 9.x и выше
hpux	HPUX 11 и выше
hpux10	HPUX 10
irix	IRIX 6 и выше
macos	MacOS 10.3 и выше

Пример применения политик для двух сегментов с ОС Windows и Linux и для всех остальных компьютеров с политикой Solaris:

```
preprocessor stream5_global: track_tcp yes
preprocessor stream5_tcp: bind_to 192.168.1.0/24,
policy windows
preprocessor stream5_tcp: bind_to 10.1.1.0/24,
policy linux
preprocessor stream5_tcp: policy solaris
```

5.4.4. Препроцессоры нормализации протоколов

Препроцессор *HTTP Inspect* предназначен для общего декодирования пользовательских приложений, использующих протокол HTTP. Он помещает полученные данные в буфер и инспектирует содержимое буфера, выделяя поля HTTP-протокола и нормализуя их в соответствии с заданными опциями. Инспекции подвергаются как запросы, так и ответы. В настоящее время анализу подвергается каждый пакет в отдельности.

Препроцессор поддерживает множество опций, которые можно использовать по умолчанию, задавая только вид профиля контролируемого HTTP-сервера.

В текущей версии реализовано три вида профилей:

1) all — нормализация URI для предупреждения известных методов атак;

2) apache — для серверов Apache. Отличается от профиля iis только использованием UTF-8 стандарта Unicode, не допуская использования обратных слэшей;

3) iis — этот профиль предназначен для серверов IIS. В этом случае используется таблица кодирования IIS Unicode и допускаются кодирование, двойное декодирование, обратные слэши и т. п. для защиты от известных атак на IIS.

Тогда следующий URI

```
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+ver
```

при применении профиля all будет представлен (нормализован) следующим образом:

```
/winnt/system32/cmd.exe?/c+ver
```

Если профиль не указан, то используется профиль по умолчанию, параметры которого приведены в табл. 5.9.

Данные таблицы показывают, что этот препроцессор позволяет обнаруживать множество известных уязвимостей.

Таблица 5.9. **Опции профиля по умолчанию**

Опция	Установленное значение
port	*80*
server flow depth	*300*
client flow depth	*300*
post depth	*-1*
chunk encoding	*alert on chunks larger than 500000 bytes*
ASCII decoding	*on, alert off*
utf 8 encoding	*on, alert off*
multiple slash	*on, alert off*
directory normalization	*on, alert off*
webroot	*on, alert on*
iis backslash	*on, alert off*
apache whitespace	*on, alert off*
iis delimiter	*on, alert off*
non strict URL parsing	*on*
max header length	*0, header length not checked*
max spaces	*200*
max headers	*0, number of headers not checked*

Формат конфигурирования препроцессора:

```
preprocessor http_inspect: \
   global \
   iis_unicode_map <map_filename> \
   codemap <integer> \
   [detect_anomalous_servers] \
   [proxy_alert] \
   [max_gzip_mem <num>] \
   [compress_depth <num>] [decompress_depth <num>] \
   [memcap <num>] \
   Disabled
```

5.4.5. Препроцессоры обнаружения аномалий

Препроцессор *sfPortscan* предназначен для обнаружения различных методов сканирования портов и использует следующие опции:

- proto { tcp udp icmp ip all } — указывает типы протоколов сканирования;
- scan_type { portscan portsweep decoy_portscan distributed_portscan all } — указывает тип сканирования, который будет обнаруживаться препроцессором;
- sense_level { low|medium|high } — задает уровень чувствительности обнаружения. Низкий уровень позволяет обнаруживать общие методы сканирования (наблюдением за ответами ошибок, такими как TCP RST или ICMP Unreachable). Средний уровень обнаруживает сканирование портов, включая фильтруемое сканирование (сканирование без получения ответов). Этот уровень используется при использовании функции NAT или кэшировании DNS-серверов. Высокий уровень имеет наименьшие пороги для обнаружения сканирования портов и значительно большее временное окно наблюдения;
- memcap { positive integer } — задает максимальный размер памяти в байтах, предназначенной для обнаружения сканирования портов;
- logfile { filename } — эта опция задает имя файла, в который будут записываться сигналы тревоги и отброшенные пакеты.

Пример:

```
preprocessor flow: stats_interval 0 hash 2
preprocessor sfportscan:\
proto { all } \
scan_type { all } \
sense_level { low }
```

5.4.6. Процессор обнаружения

Основой работы механизма обнаружения является сравнение с имеющимися правилами. Этот компонент Snort принимает данные от декодера пакетов и препроцессоров (если они активированы) и сравнивает значения полей этих данных с правилами, заданными в файле конфигурации. Процессор обнаружения сначала пытается определить, какой набор правил следует использовать для конкретного фрагмента данных. В первую очередь это определяется по соответствующему протоколу (TCP, UDP, ICMP или IP), а затем идентифицируются характеристики в рамках соответствующего протокола. Для TCP и UDP — это номера портов источника и назначения, для ICMP — это тип сообщения.

При сравнении с правилами обычно используется вариант, когда Snort работает по принципу первого совпадения — срабатывает

первое правило, условия которого удовлетворены. В текущую версию Snort включена возможность выполнять несколько сравнений для одного события и генерировать несколько сигналов тревоги для одного пакета. Кроме того, существует возможность выбора последовательности применения правил, связанная с принципиальной возможностью атак обмана СОВ.

Для реагирования на множественные сигналы тревог предусмотрена возможность сигнализации об определенном количестве появления некоторого набора данных в пределах заданного интервала времени. Эта возможность называется пороговой классификацией. При этом можно выбрать следующие варианты: выдавать сигнал тревоги после получения первых *n*-сигналов о данном событии, или на каждые *n*-экземпляров данного события.

5.4.7. Модули вывода

Модули вывода позволяют администратору формировать и представлять выходные данные Snort. Модули вывода включаются в работу всякий раз, когда вызываются подсистемы регистрации и сигнализации, после завершения работы декодера и препроцессоров.

Модули вывода можно рассматривать как модули расширения системы, так как они могут быть написаны пользователями системы и включены в Snort в процессе компиляции.

В каждом из модулей вывода можно выделить семь ключевых аспектов: информация о заголовках и копирайте, include-файлы, зависимости (dependencies) и глобальные переменные, регистрация ключевых слов, анализ аргументов и включение в список функций, форматирование, обработка и хранение данных, препроцессорная обработка, очистка областей памяти и завершение работы приложения.

Snort предлагает пользователю несколько способов регистрации как генерируемых сигналов тревоги, так и связанных с ними данных пакетов. Эти данные могут регистрироваться в коде ASCII или в двоичном формате и записываться с помощью различных модулей вывода.

Записи в двоичном формате могут быть восстановлены с помощью любого анализатора пакетов, например Ethereal, TCPdump или IRIS.

Для текущей версии Snort разработаны модули, позволяющие посылать сигналы тревоги в виде запросов SNMP (Simple Network Management Protocol) удаленному серверу SNMP, а также модуль вывода SMB Alerting, посылающий сигналы тревоги удаленным Windows-системам в реальном масштабе времени.

Реализованы модуль регистрации в формате PCAP (Packet Capture Library), модуль регистрации согласно стандарту XML IDMEF. Кро-

ме того, Snort имеет возможность записывать сигналы тревоги и пакеты в различные базы данных, включая MySQL, PostgreSQL, SQL Server и Oracle.

5.4.8. Правила Snort

Одной из самых сильных сторон Snort является возможность для пользователя разрабатывать и использовать свои собственные правила. Для построения правил используется простой язык описаний, с помощью которого можно написать собственные правила, исходя из специфики применения Snort. Большинство правил записывается в одной строке. В текущей версии допускается написание правил, занимающих несколько строк, при этом в конце каждой строки пишется символ обратного слэша (\).

Правило Snort состоит из двух основных частей: заголовка правила и опций правила. Заголовок правила содержит действие правила, протокол, адреса и маски IP-адресов источника и назначения, номера портов источника и назначения. Опции правила содержат сообщения, выдаваемые при генерации тревоги, и информацию о том, какую часть пакета необходимо просматривать для сравнения с правилом. Опции правила заключаются в скобки. Пример простого правила (для наглядности правило записано двумя строками с использованием знака \):

```
alert tcp any any -> 192.168.1.0/24 111 \
(content: "|00 01 86 a5|"; msg: "mountd access";)
```

В этом случае текст до открывающей скобки представляет собой заголовок правила. Часть правила, заключенная в круглые скобки, содержит опции правила. Имя опции (ключевое слово) заканчивается двоеточием, после которого следует содержимое опции, которое, в свою очередь, заканчивается точкой с запятой.

Заголовок правила. Первым элементом заголовка является действие, которое указывает Snort, что делать при совпадении данных пакета с опциями правила и перечень действий, которые можно указывать в заголовке правила:

- alert — генерировать сигнал тревоги, используя заданный метод, и записать пакет в журнал;
- log — записать пакет в журнал;
- pass — игнорировать пакет;
- activate — генерировать сигнал тревоги и активировать динамическое равило;
- dynamic — действие данного правила игнорируется до активации правилом активации (опцией activate), после чего записывается в журнал;

• drop — посредством iptables удалить пакет и записать его в журнал;

• reject — посредством iptables удалить пакет, записать его в журнал и послать пакет TCP RST (для протокола TCP) или ICMP port unreachable (для протокола ICMP);

• sdrop — позволить iptables удалить пакет, но не записывать его в журнал.

Следующим полем заголовка является протокол. В текущей версии Snort для анализа подозрительного поведения используются четыре протокола: TCP, UDP, ICMP и IP. Планируется введение следующих протоколов: ARP, IGRP, GRE, OSPF, RIP, IPX и др.

Очередными полями заголовка являются адрес и порт. Может использоваться ключевое слово *any* для определения любого адреса. Адрес может сопровождаться блоком (CIDR, Classless Inter-Domain Routing), обозначающим маску. Блок маски /24 показывает на сеть класса C, блок /16 на сеть класса B, а блок /32 на определенный адрес (например, комбинация 192.168.1.1/24 указывает на блок адресов от 192.168.1.1 до 192.168.1.255).

Предусмотрен оператор отрицания — восклицательный знак. Этот оператор указывает Snort рассматривать все адреса, за исключением отмеченных оператором отрицания.

В заголовке правила можно указать диапазон адресов, который заключается в квадратные скобки:

```
alert tcp ![192.168.1.0/24, 10.1.1.1/24] any -> \
[192.168.1.0/24, 10.1.1.1/24] 111 \
(content: "|00 01 86 a5|"; msg: "mountd access";)
```

Очередным полем заголовка является номер порта. Порт может быть задан множеством способов: ключевое слово *any* указывает на любой номер порта, просто номер порта (в примере — 111), диапазоном номеров (например, 8000:8080). При задании номера порта можно также использовать оператор отрицания (за исключением применения оператора отрицания к ключевому слову *any*).

Далее в заголовке Snort используется оператор направления —>, показывающий направление трафика, к которому необходимо применить правило. Если необходимо применять правило к двунаправленному трафику, используется символ <>. Символ направления <— не используется, так как оно заменяется символом —> с соответствующей заменой адресных частей заголовка.

После оператора направления указываются адрес, маска и порт назначения.

Опции правил. Опции правил являются основой механизма обнаружения, предоставляя возможности для обнаружения. Существует четыре категории опций:

• meta-data — предоставление информации о правиле, которая не используется в процессе обнаружения;

• payload — обеспечивает просмотр данных внутри пакета;

• non-payload — просмотр данных, не содержащихся в нагрузке пакета;

• post-definition — указывает на специфические переходы, выполняемые при срабатывании правила.

Далее кратко рассмотрим опции правил по категориям.

Опции meta-data. К опциям этой категории относятся опции: msg, reference, sid, rev, classtype и priority:

• msg — указывает механизмам регистрации и генерации тревоги содержимое соответствующего сообщения, которое задается в виде текстовой строки, заключенной в кавычки, например (msg: «IMAP buffer overflow»);

• reference — позволяет сделать ссылку на внешние системы идентификации атаки. Поддерживаются следующие ссылки: bugtraq (http://www.securityfocus.com/bid/), cve (http://cve.mitre.org/cgi-bin/cvename.cgi?name=), nessus (http://cgi.nessus.org/plugins/dump.php3?id=), arachnids (http://www.whitehats.com/info/IDS), mcafee (http://vil.nai.com/vil/dispVirus.asp?virus_k=), url (http://);

• sid — используется для уникальной идентификации правил Snort. В настоящее время все sid разделены на группы: до 100 — зарезервированы, от 100 до 1 000 000 — включаются в дистрибутив Snort, более 1 000 000 — для локального применения;

• rev — позволяет перезаписывать сигнатуры и их описания с новой информацией (используется совместно с sid);

• classtype — характеризует категорию тревоги (указывает класс атаки). Пользователь может определить приоритет для каждого типа правил. Существующие категории и их приоритеты находятся в файле classification.config;

• priority — приоритет устанавливает уровень важности правила, например в теле правила можно указать: (content: «/cgi-bin/phf»; priority:10;).

Опции обнаружения содержимого (payload detection). Эти опции являются наиболее значимыми и интересными. К числу этих опций относятся: content, uricontent, iisdataat, pcre и др.:

• content — позволяет использовать правило, которое ищет определенные данные в содержимом пакета. Данные могут представлять собой текстовую строку или двоичные данные. Например, content:!"GET" (представляет текстовую строку), content: "|5c 00|P|00|I|00|P|00|00 5c|"; (представляет смесь текстовых и двоичных данных).

Для ключевого слова content, в свою очередь, можно указать модифицирующие ключевые слова: depth, offset, distance, within, nocase и rawbytes, которые предоставляют различные варианты возможностей поиска заданных строк: depth — определяет количество байтов, которые данное правило должно анализировать при поиске заданного содержимого; offset — указывает препроцессору, что по-

иск заданной строки надо начинать с байта offset; distance — указывает количество байтов, которые должны игнорироваться до начала сравнения; within — позволяет удостовериться, что заданное опцией количество байтов находится между указанными образцами, например, (content:«ABC»; content:«EFG»; within:10;); nocase — указание на игнорирование регистра текста в содержании правила, например, (content:«USER root»; nocase;); rawbytes — это ключевое слово позволяет Snort рассматривать содержимое пакета в шестнадцатеричном представлении, не учитывая тип кодирования, например, (content: «|FF F1|»; rawbytes;); uricontent — эта опция позволяет анализировать трафик запрашивающей системы только в URI-разделе запроса, нормализуя найденные строки, например URI:

```
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+ver и
/cgi-bin/aaaaaaaaaaaaaaaaaaaaaaaaa/..%252fp%68f?
```

будут нормализованы в

```
/winnt/system32/cmd.exe?/c+ver и /cgi-bin/phf?
```

Опция iisdataat определяет, что содержимое находится в определенном месте, соответствуя концу данных, полученных в предыдущем пакете. Опция pcre позволяет записать правило, используя язык Perl.

Учитывая значительное число атак, использующих специфику реализации стека TCP\IP, Snort содержит большое число опций, которые не связаны с содержимым пакета. Рассмотрим некоторые из этих опций, задаваемых следующими ключевыми словами:

- fragoffset — позволяет сравнивать значение поля offset фрагмента с заданным значением, например для просмотра первых фрагментов (fragbits: M; fragoffset: 0;);
- ttl — позволяет проверять время жизни пакета, например (ttl: <3;);
- tos — проверяет поля TOS на соответствие заданному значению, например (tos:!4;);
- id — позволяет сравнить значение поля ID IP-пакета с заданным значением, например (id:31337;);
- ipopts — используется для проверки наличия определенных опций IP. Контролю могут подвергаться следующие опции: rr — Record route, eol — End of list, nop — No operation, ts — Time Stamp, sec — IP security option, lsrr — Loose source routing, ssrr — Strict source routing, satid — Stream identifier, any — установка любой опции. В правиле может содержаться только одно ключевое слово ipopts, например (ipopts:lsrr;);
- fragbits — позволяет контролировать биты фрагментации и зарезервированные биты, для чего можно использовать обозначения битов (M — есть еще фрагменты, D — запрет фрагментирования,

R — зарезервированные биты) и простые операции сравнения (знак «+» означает соответствие заданного бита и любые значения других битов, знак «*» означает срабатывание, если любой из заданных битов установлен, знак «!» вызовет срабатывание, если ни один из указанных битов не установлен). Например, опция (fragbits:MD+;) вызовет срабатывание, если будут установлены биты More fragments и Do not Fragment;

- dsize — позволяет контролировать длину пакета, например (dsize:300<>400;);
- flags — позволяет контролировать установку флагов TCP (F — FIN, S — SYN, R — RST, P — PSH, A — ACK, U — URG, 1 — зарезервированный бит 1, 2 — зарезервированный бит 2, 0 — отсутствие флагов TCP) и простые операции сравнения (знак «+» означает соответствие заданного бита и любые значения других битов, знак «*» означает срабатывание, если любой из заданных битов установлен, знак «!» вызывает срабатывание, если ни один из указанных битов не установлен). Например, для контроля наличия флагов SYN и FIN и игнорирования зарезервированных битов можно задать опцию в виде (flags:SF,12;).

Кроме того, возможны проверки значений и других параметров пакетов, для чего служат следующие опции: seq (последовательный номер TCP-пакета, ack (значение подтверждения), window (значение окна передачи TCP), itype (тип сообщения ICMP), icode (код сообщения ICMP), icmp_id (значение идентификатора ICMP), icmp_seq (последовательный номер ICMP-сообщения), ip_proto (имя протокола в заголовке IP), sameip (проверка равенства адреса источника и адреса назначения) и др.

Рассмотренные опции позволяют реализовать достаточно сложные правила обнаружения различных атак и их вариантов.

5.4.9. Примеры правил

В качестве примера применения рассмотренных опций рассмотрим несколько реальных правил Snort.

Правило для определения TCP-пакета, в котором установлен только флаг FIN. Такие пакеты используются атакующими для определения операционной системы целевого хоста, поскольку машины с ОС Windows отвечают на него пакетом с установленными флагами ACK и RST (независимо от того, открыт или закрыт соответствующий порт), а системы Unix отвечают пакетом с такими же флагами только в случае, когда порт закрыт. Соответствующее правило имеет следующий вид:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any \
(msg:»SCAN FIN»; flags: F; reference:arachnids,27; \
classtype:attempted-recon; sid:621; rev:1;)
```

Рассмотрим правило, определяющее поведение атакующего после получения доступа к Windows-серверу путем использования IIS-уязвимости, когда атакующий пытается использовать командный интерпретатор cmd.exe. Затем атакующий пытается просматривать директорию, используя команду dir. Строка «Volume Serial Number» обычно содержится в списке директорий для Windows NT\2000\XP:

```
alert tcp $HTTP_SERVERS $HTTP_PORTS -> $EXTERNAL_NET  \
any (msg:»ATTACK RESPONSES http dir listing»;         \
content: «Volume Serial Number»;                      \
flow:from_server,established; classtype:bad-unknown;\
sid:1292; rev:4;)
```

Следующее правило предназначено для обнаружения сетевого червя Slammer, пытающегося использовать уязвимость переполнения буфера в Resolution Service MS SQL Server 2000:

```
  alert udp $EXTERNAL_NET any -> $HOME_NET 1434      \
(msg:»MS-SQL Worm propagation attempt»;              \
  content:»|04|»; depth:1;                           \
  content:»|81 F1 03 01 04 9B 81 F1 01|»;            \
  content:»sock»; content:»send»;                    \
  reference:bugtraq,5310; classtype:misc-attack;     \
  reference:bugtraq,5311;                            \
  reference:url,vil.nai.com/vil/content/v_99992.htm;\
sid:2003; rev:2;)
```

5.5. Обнаружение аномалий

Основой для обнаружения аномалий (ОА) является понятие «нормального» поведения. Тогда любые отклонения от нормальности система рассматривает как аномалию. Поэтому системы обнаружения аномалий могут различаться как в зависимости от используемого подхода к определению нормальности, так и в зависимости от используемого подхода к определению отклонений от нормальности.

Обнаружение аномалий обычно включает в себя процесс установления профилей нормального поведения пользователей, сравнение действительного поведения с этими профилями и сигнализацию об отклонениях от нормального поведения. Основной гипотезой при обнаружении аномалий является утверждение, что образцы ненормального поведения ассоциируются со злоупотреблением системы. Профили определяются как множества метрик, которыми являются измерения частных аспектов поведения системы или пользователя. Для каждой метрики предусматривается или порог, или диапазон разрешенных значений.

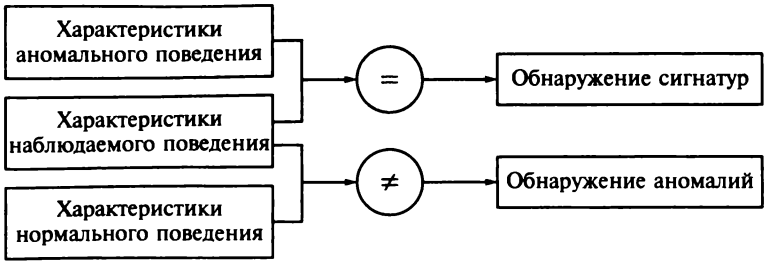


Рис. 5.6. Схема принятия решений при использовании технологий обнаружения аномалий и сигнатур

Обнаружение аномалий зависит от предположения, что пользователи выполняют предсказуемые последовательные действия. Этот подход допускает адаптацию к изменениям поведения пользователей во времени. Полноту метода обнаружения аномалий необходимо контролировать (устанавливать), поскольку никто не знает, является ли данное множество метрик достаточным для представления всего множества возможного аномального поведения. Поэтому в каждом случае требуются дополнительные исследования для удостоверения в том, что обнаружение аномалий будет способно обнаружить все сценарии.

Существенной особенностью обнаружения аномалий является необходимость установления «нормального» поведения.

Схема принятия решения при использовании технологий обнаружения аномалий и сигнатур приведена на рис. 5.6.

Некоторые современные СОВ нельзя отнести ни к системам обнаружения сигнатур, ни к системам обнаружения аномалий. Они опираются на новые (иногда их называют альтернативные) подходы к обнаружению. К числу таких альтернативных подходов можно отнести:

- методы Data Mining;
- методы технологии мобильных агентов;
- методы построения иммунных систем;
- применение генетических алгоритмов;
- применение нейронных сетей.

Рассмотрим примеры использования данных подходов к построению систем обнаружения.

5.5.1. Методы Data Mining

Data Mining переводится как «добыча» или «раскопка данных». Возникновение этого термина связано с новым шагом в развитии средств и методов обработки данных.

В основу современной технологии Data Mining (discovery-driven data mining) положена концепция шаблонов, отражающих фрагменты многоаспектных взаимоотношений в данных. Эти шаблоны представляют собой закономерности, свойственные подвыборкам данных, которые могут быть компактно выражены в понятной человеку форме. Г.Пиатецкий-Шапиро (один из основателей этого направления) определил технологию Data Mining как процесс обнаружения в необработанных (raw) данных:

- ранее неизвестных;
- нетривиальных;
- практически полезных;
- доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности.

Data Mining является мультидисциплинарной областью, возникшей и развивающейся на базе достижений прикладной статистики, распознавания образов, методов искусственного интеллекта, теории баз данных и др. Отсюда обилие методов и алгоритмов, реализующих в различных действующих системах Data Mining. Многие из таких систем интегрируют в себе сразу несколько подходов.

Рассмотрим основные классы систем Data Mining.

Наиболее широким подклассом *предметно-ориентированных аналитических систем*, получившим распространение в области исследования финансовых рынков, является «технический анализ», который представляет собой совокупность множества методов прогнозирования динамики цен и выбора оптимальной структуры инвестиционного портфеля, основанных на различных эмпирических моделях динамики рынка.

Последние версии статистических программных пакетов включают в себя наряду с традиционными *статистическими методами* и элементы Data Mining.

Нейронные сети — большой класс систем, архитектура которых имеет аналогию (довольно слабую) с построением нервной ткани из нейронов.

Системы рассуждений на основе аналогичных случаев (CBR, case based reasoning) для прогноза на будущее или выбора правильного решения ищут в прошлом близкие аналоги ситуации и выбирают тот ответ, который был для них правильным.

Деревья решений (decision trees) создают иерархическую структуру классифицирующих правил типа if—then, имеющую вид дерева.

Рассмотрим *эволюционное программирование*. Гипотезы о виде зависимости целевой переменной от других переменных формулируются в виде программ на некотором внутреннем языке программирования. Процесс построения программ строится как эволюция в мире программ. Когда система находит программу, более или менее удовлетворительно выражающую искомую зависимость, она начинает вносить в нее небольшие изменения и отбирает среди построенных

таким образом программ-потомков те, которые повышают точность. В результате проведения подобной эволюции система получает несколько генетических линий программ, которые конкурируют между собой в точности выражения искомой зависимости.

Другое направление эволюционного программирования связано с поиском зависимости целевых переменных от остальных в форме функций определенного вида. Например, в методе группового учета аргументов зависимость ищется в форме полиномов.

Хотя *генетические алгоритмы* используются в основном как мощное средство решения разнообразных комбинаторных задач и задач оптимизации, они в последнее время включаются в стандартный инструментарий методов Data Mining.

Алгоритмы ограниченного перебора, предложенные в середине 60-х годов XX в. М. М. Бонгардом, вычисляют частоты комбинаций простых логических событий в подгруппах данных. На основании анализа вычисленных частот делается заключение о полезности той или иной комбинации для установления ассоциации в данных, классификации, прогнозирования и т. д.

Указанные методы в последнее время начинают применяться и для решения задач обнаружения атак и вторжений.

5.5.2. Методы технологии мобильных агентов

Для получения полной картины о состоянии безопасности в сетях организации некоторые СОВ используют распределенный сбор данных (и определенных функций предварительной обработки) от контролируемых хостов, но функции анализа и принятия решения возлагаются при этом на единственный механизм.

Такой подход к организации распределенного обнаружения имеет свои недостатки:

- анализатор (устройство принятия решения) — единственная точка отказа. Если нарушитель выведет ее из строя или обойдет, то система обнаружения не выполнит своих функций;
- ограниченная расширяемость, так как обработка данных в единственной точке ограничивает число контролируемых элементов сети организации;
- сложность с конфигурированием и реконфигурированием СОВ, особенно при добавлении новых функций в СОВ;
- возможность обмана системы, поскольку она отличается от защищаемой системы.

Поэтому в последние годы разрабатываются варианты распределенного сбора данных (сенсоров) и распределенного анализа данных (принятия решения). Одной из технологий, которая может быть использована для построения распределенных СОВ, является технология агентов.

В научной и периодической литературе используются термины: агенты, автономные агенты, программные агенты, мобильные агенты и т. д. Под автономным агентом (или просто агентом) понимается программный агент, который осуществляет определенные функции контроля хоста. Под автономностью агента подразумевается то, что он является независимо исполняемым объектом (его выполнение контролируется только ОС, а не другими объектами). Агенты могут получать данные от других объектов системы или агентов, кроме того, агенты могут получать определенные команды.

Поскольку агенты являются независимо исполняемыми объектами, они могут быть добавлены или удалены из системы без уведомления других компонент, т. е. без необходимости рестарта СОВ. Агент может быть частью группы агентов, которые выполняют разные функции, и использовать данные, получаемые другими агентами.

Поэтому использование технологии агентов для сбора и анализа данных имеет следующие достоинства:

- каждый агент можно запрограммировать на получение данных из своего источника (записи аудита, данные опроса системы, перехват сетевых пакетов и др.). Это позволяет совместно использовать параметры, характерные как для хостовых, так и для сетевых СОВ;
- возможность независимого старта и остановки работы различных агентов позволяет оперативно в ходе работы СОВ изменять содержание (тип) агентов и их количество;
- агенты могут быть программами, написанными на различных языках программирования (тех, которые наиболее приспособлены для решаемой агентом задачи).

Использование автономных агентов для обнаружения вторжений послужило толчком к разработке различных прототипов СОВ, использующих технологию агентов. В качестве примера рассмотрим архитектуру системы AAFID (Autonomous Agent For Intrusion Detection), разработанную в университете Пурду.

Основными компонентами системы являются: агенты, анализаторы и мониторы. Пример такой архитектуры для трех хостов приведен на рис. 5.7.

Агент — независимо исполняемый объект, который контролирует определенные аспекты хоста и докладывает об обнаруженных аномалиях или «интересном» поведении соответствующему анализатору.

Каждый хост защищаемой системы может содержать произвольное число агентов, контролирующих возникающие на хосте события. Все агенты хоста представляют данные одному анализатору. Агент не может генерировать сигнал тревоги.

Анализатор — внешний интерфейс связи каждого хоста. Он контролирует все операции агентов на своем хосте и может останавливать их работу, запускать агентов и выдавать им команды. Кроме того, анализатор выполняет функции редукции объема данных, предоставляемых агентами. Результаты своей работы анализатор вы-

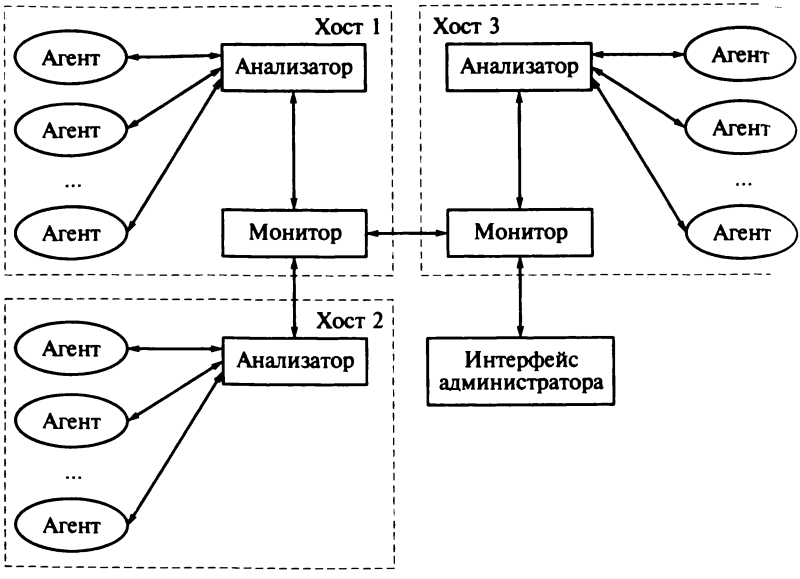


Рис. 5.7. Пример архитектуры AAFID для трех хостов

дает монитору (одному или более). Можно выделить две основные функции анализатора: управление (запуск и остановка исполнения агента на хосте, сохранение записей о деятельности агента, ответ на полученные команды от монитора) и обработка (получение генерируемых агентами данных, обработка и анализ полученных данных, предоставление данных монитору).

Монитор — высокоуровневый компонент архитектуры AAFID. Он также выполняет функции управления (в режиме управления мониторы могут получать команды от других мониторов и выдавать команды мониторам и анализаторам) и обработки (монитор обрабатывает данные, полученные от анализаторов, поэтому может обнаруживать вторжения, которые не были обнаружены анализаторами).

Кроме того, мониторы имеют возможность взаимодействовать с интерфейсом администратора, предоставляя администратору возможность доступа ко всем элементам системы.

Схема иерархического взаимодействия элементов системы AAFID приведена на рис. 5.8.

Таким образом, архитектура AAFID позволяет осуществлять сбор данных от множества источников, распределенных по защищаемой системе, и дает возможность комбинировать достоинства характеристик обнаружения хостовых и сетевых СОВ. Модульность архитектуры позволяет расширять систему, легко ее конфигурировать и модифицировать, что позволяет использовать следующие возможности (не реализованные в самом проекте AAFID):

• обучение агентов с использованием различных методов машинного обучения;

• эволюцию агентов во времени с использованием генетического программирования;

• сохранение состояния контролируемых сеансов между сеансами, что позволит обнаруживать атаки, протяженные во времени;

• мобильность агентов, т.е. возможность их перемещения с хоста на хост при комбинировании AAFID с другими агентскими архитектурами.

Одним из существенных недостатков рассмотренных распределенных СОВ является их уязвимость к атакам против самой системы обнаружения. Компоненты распределенной СОВ находятся в статическом положении и связаны друг с другом иерархической структурой, что позволяет атакующему вывести СОВ из строя, воздействуя на один из высших уровней иерархии. Одним из вариантов противодействия таким атакам является подход на основе мобильных агентов. При данном подходе неподвижными остаются только листья иерархического дерева, а внутренние узлы дерева являются мобильными агентами, так как они не имеют непосредственного взаимодействия с источниками данных хоста или трафика сети. Поскольку внутренние узлы получают данные от других компонентов, обрабатывают их и передают результаты вышележащим узлам, эта обработка может осуществляться в любом месте сети.

Если эти узлы перемещаются по сети случайным образом, то это затруднит атакующему определение их местоположения. Если атакующий выведет из строя хост (применив, например, атаку отказа в обслуживании), то агенты могут переместиться в другое место, которое,

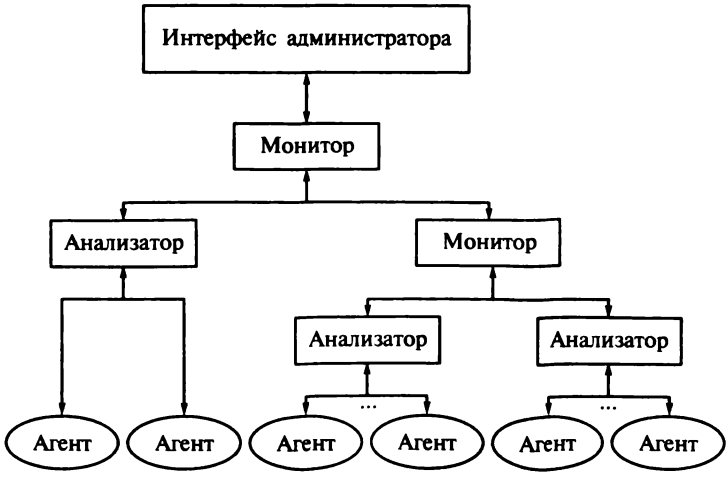


Рис. 5.8. Схема иерархического взаимодействия элементов AAFID

«по их мнению», является безопасным. Если агенты уничтожены, то есть возможность «воскресить» и подсоединить их к существующей структуре СОВ.

СОВ могут функционировать как отдельно стоящее, централизованное или интегрированное приложения, которые создают распределенную систему. Последняя имеет частную архитектуру с автономными агентами, которые могут осуществлять предварительные вычисления, реализовывать реакции и даже перемещаться по сети.

5.5.3. Методы построения иммунных систем

При построении подобных систем используется аналогия между биологическими иммунными системами и механизмами защиты компьютерных систем. Ключевым параметром для успешного функционирования обеих систем является способность формировать определение «свое-чужое», т.е. способность иммунной системы организма определять, какие материалы являются безопасными сущностями (материалы самого организма) и какие являются патологическими или другими опасными факторами. По аналогии с биологическими организмами, в которых подобная способность базируется на коротких протеиновых фрагментах, исследователи сфокусировались на некоторых компьютерных атрибутах, которые можно рассматривать как их аналоги. Группа исследователей выдвинула гипотезу, что последовательности системных вызовов операционной системы Unix могут удовлетворять этим требованиям. Для реализации идеи они ограничились короткими последовательностями системных вызовов (даже без учета параметров, передаваемых в вызовах) и их временной последовательностью.

Сначала исследовательская система, построенная по этому принципу, предназначалась для обнаружения аномалий, хотя она также могла обнаруживать злоупотребления. Построение такой системы осуществляется в две фазы.

На первой фазе строится база знаний, в которую помещаются профили нормального поведения. Считается, что характеристики анализируемого поведения не являются зависящими от пользователя (user-centric), а зависят от активности процессов. Тогда отклонения характеристик поведения от этих профилей считаются аномалиями.

Во второй фазе профили используются для контроля последовательного поведения системы на аномальность. Последовательности системных вызовов, являющиеся результатом выполнения привилегированных процессов в системе, собираются с учетом времени. Эти профили содержат уникальные последовательности длиной 10 (данная величина выбрана исследователями на основе опыта). Для характеристики отклонения от нормального поведения используются три

значения: удачное использование уязвимости, неудачное использование уязвимости и ошибочное условие.

Результаты экспериментов оказались весьма обнадеживающими, поскольку данные три значения измерений, предназначенные для обнаружения некоторых видов аномального поведения, покрывают несколько давно известных проблематичных Unix-программ. Кроме того, исследования показали, что множества выполняемых последовательностей компактны.

Хотя данный подход показал свою мощность, он не является полным решением проблемы обнаружения вторжений. Некоторые атаки (включая ошибки синхронизации, маскарад и нарушения политики безопасности) не используют привилегированные процессы. Поэтому такие атаки не могут быть обнаружены с помощью данного подхода.

5.5.4. Применение генетических алгоритмов

Генетические алгоритмы — представители класса алгоритмов, называемых эволюционными. Эволюционные алгоритмы опираются на концепцию естественного отбора Ч.Дарвина для оптимального решения проблем. Идея генетического алгоритма (ГА) заимствована у живой природы и основана на принципах эволюции и естественного отбора. Генетический алгоритм был разработан Голландом (John Holland) в Мичиганском университете в 1975 г. Голдберг (David Goldberg) выдвинул ряд гипотез и положений, помогающих глубже понять природу ГА. В общем случае ГА использует подход, наблюдаемый в живой природе: естественный отбор, приспособляемость к изменяющимся условиям среды, наследование потомками жизненно важных свойств от родителей и т. д.

Суть ГА состоит в следующем. Фиксируется начальная популяция — множество наборов решений задачи, которые достаточно далеки от точного решения. Для каждого члена популяции вычисляется значение функции «согласия» с решением. Одним из вариантов является функция средней ценности популяции.

Если на шаге k алгоритма имеется популяция $G(k)$, состоящая из N строк S_i^k, то средняя ценность популяции может быть определена следующим образом:

$$F_{\text{ср}}[G(k)] = 1/N \sum F(S_i^k),\ i = 1, 2, \ldots, N.$$

Генетический алгоритм осуществляет переход от популяции $G(k)$ к популяции $G(k + 1)$ таким образом, чтобы средняя ценность составляющих ее строк увеличивалась. Можно рассматривать случай, когда количество элементов (строк) в новой популяции $N(k + 1) = \lambda N(k)$, где λ — коэффициент новизны популяции. Если $\lambda < 1$, то популяция будет перекрывающейся (в новой популяции сохраняются

некоторые строки из старой). Если $\lambda = 1$, то популяция будет не перекрывающейся, т.е. подвергнется полному обновлению.

Алгоритм состоит в последовательном выполнении ряда шагов до получения решения. На каждом шаге работы ГА к членам популяции (называемым хромосомами) применяются операторы селекции, скрещивания и мутации. Названия этих операторов заимствованы из живой природы.

Оператор селекции — для формирования новой популяции попарно выбираются члены текущей популяции с максимальным значением целевой функции (точнее, с вероятностью, определяемой степенью близости). Операторы селекции строятся таким образом, чтобы с ненулевой вероятностью любой элемент популяции мог бы быть выбран в качестве «родителя». Иногда допускается случай, когда один член популяции является родителем один и родителем два.

Оператор скрещивания — по имеющимся членам популяции k_i и k_j строится новое решение k_m. Существует множество версий этого оператора. Одним из простейших является однородный оператор, который по членам k_i и k_j строит решение k_m, присваивая каждой координате этого члена с вероятностью 0,5 соответствующее значение одного из родителей. Поэтому если какая-то координата у родителей совпадает, то новый элемент будущей популяции унаследует это значение. Возможен случай, когда для получения нового элемента используется более двух «родителей».

Оператор мутации — полученное решение k_m подвергается случайной модификации. Обычно модификация реализуется заменой значения каждой координаты на противоположное с заданной вероятностью. Полученная после применения этих операторов популяция является основой для дальнейшей работы.

Общая схема генетического алгоритма показана на рис. 5.9.

Генетической алгоритм работает с представленными в конечном алфавите строками S конечной длины n, которые используются для кодирования некоторого множества альтернатив W.

Каждая строка представляет собой упорядоченный набор из n элементов: $S = \{s_1, s_2, ..., s_n\}$, каждый из которых может быть задан в своем собственном алфавите V_i, $i = 1, 2, ..., n$.

Для работы ГА необходимо на множестве строк задать неотрицательную функцию $F(S)$, определяющую «ценность» строки S. Алгоритм производит поиск строки, для которой $F^*(S) = \text{argmax}F(S)$. Если на множестве W задана целевая функция $f(w)$, то $F(S)$ может быть определена следующим образом:

$$F(S) = f(w),$$

если элемент w при отображении исходного множества W на множество строк был сопоставлен строке S.

Популяция $G(k)$ на шаге k представляет собой конечный набор строк $G(k) = (s_1^k, s_2^k, ..., s^k_N)$, где N — размер популяции. Отметим, что строки в популяции могут повторяться.

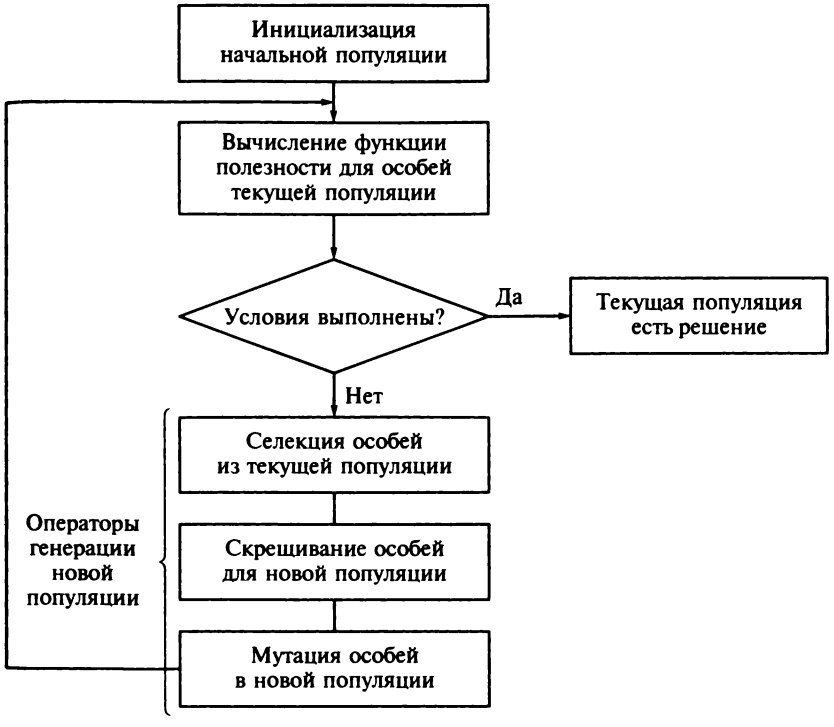


Рис. 5.9. Общая схема генетического алгоритма

Рассмотрим пример, поясняющий работу операторов ГА.

Пусть в качестве родителей выбраны следующие хромосомы (N = = 16, для наглядности заменим нули и единицы на буквы):

```
(a,b,b,b,a,b,b,a,a,a,a,b,b,b,a,a)
(x,y,y,x,x,x,y,x,x,y,y,x,x,y,x,y)
```

Выбираем номер позиции скрещивания (считая, например, все позиции равновероятными). Пусть это будет 10. Тогда после применения скрещивания получим следующие хромосомы для новой популяции:

```
(a,b,b,b,a,b,b,a,a,y,y,x,x,y,x,y)
(x,y,y,x,x,x,y,x,x,a,a,b,b,b,a,a)
```

Таким образом, для первой хромосомы все гены на позициях 1—9 останутся, а гены на позициях с 10 по 16 заменятся соответствующими генами второй хромосомы. Во второй хромосоме останутся гены на позициях 1—9, а гены позиций 10—16 заменятся генами первой хромосомы.

При применении оператора мутации значение каждого гена полученной в результате скрещивания хромосомы с заданной вероятностью заменяется на противоположное (обычно выбирается вероятность, значительно меньшая единицы, например 0,001).

Процесс обнаружения вторжений включает в себя определение векторов гипотез для тех событий, в которых вектор может представлять собой вторжение. Гипотезы затем проверяются на их корректность. На основе результатов проверки гипотезы улучшаются. Этот процесс повторяется до нахождения решения. Роль генетических алгоритмов в этом процессе состоит в образовании улучшенных гипотез.

Схема анализа на основе генетических алгоритмов включает в себя две стадии:

- кодирование решений проблемы строкой бит;
- нахождение значений функции пригодности (fitness) для проверки (тестирования) каждого индивидуума популяции (например, всех возможных решений проблемы), удовлетворяющей некоторому эволюционному критерию.

В системе GASSATA генетические алгоритмы применяются к проблеме классификации системы событий с использованием множества векторов гипотез $\{h\}$ (один вектор на поток интересующих систему событий) размерностью n (n — число потенциально известных атак). По определению $h = 1$, если оно представляет атаку, и $h = 0$, если атака отсутствует.

Функция пригодности содержит риск того, что частная атака присутствует в системе. Этот риск умножается на значение вектора гипотезы, и полученное произведение корректируется в соответствии с квадратичной функцией штрафов для выделения нереальных гипотез. Данный шаг улучшает различимость атак. Цель процесса состоит в оптимизации результатов анализа так, чтобы вероятность обнаружения реальной атаки приближалась к единице, а вероятность ошибки в обнаружении атаки стремилась к нулю.

В проведенных экспериментах средняя вероятность точного обнаружения реальной атаки составила 0,996, а средняя вероятность ложных срабатываний — 0,0044. Время, требуемое для настройки фильтров (функции пригодности), было незначительным. Множество из 200 атак потребовало для системы 10 мин 25 с для оценки записей аудита, генерируемых средним пользователем за 30 мин интенсивного использования системы.

Рассмотрим один из примеров применения ГА к обнаружению вторжений, когда ГА применяется для создания простых правил поиска атак в сетевом трафике. Эти правила используются, чтобы отличить нормальные сетевые соединения от аномальных. Аномальные события вероятно связаны с вторжением. Правила хранятся в базе правил в следующей форме:

```
if (условие) then (действие)
```

Условие означает совпадение между текущим соединением и правилом СОВ (например, совпадение адресов источника и назначения, номеров портов, продолжительности соединения, используемого протокола и т.д.). Действие определяется ПБ организации (например, генерация сигнала тревоги администратору, прекращение соединения, запись в файл аудита или все эти действия вместе).

Правило для определения аномального соединения может быть следующим:

```
if {соединение содержит следующую информацию:
   адрес источника: 10.0.0.15;
   адрес назначения: 192.168.0.5;
   порт назначения: 555;
   продолжительность соединения: 7,5 с}
then {разорвать соединение}
```

Конечной целью применения ГА является генерация правил, которые определяют только аномальные соединения. Эти правила тестируются на имеющихся архивных записях сеансов и используются для фильтрации новых сетевых соединений, чтобы определить подозрительный (аномальный) сетевой трафик.

В качестве исходных данных выбираются те поля сетевых пакетов, которые связаны с соединениями. Для простоты рассмотрим ограниченное число атрибутов (полей), которые представлены в табл. 5.10.

Тогда правило для данных таблицы выглядит следующим образом:

```
if {соединение: адрес источника: 10.0.?.?;
адрес назначения: 192.168.176+?.?;
порт источника: 33337; порт назначения: 80;
время соединения: 482 с;
соединение закончено инициатором; протокол TCP;
инициатор послал: 7 320 байт;
приемник получил: 38 891 байт }
then { разорвать соединение }
```

Преобразуем данное правило в форму хромосомы (для наглядности вставим пробелы, разделяющие поля):

```
(0,a,0,0,-1,-1,-1,-1, c,0,a,8,b,-1,-1,-1,
3,3,3,3,7, 0,0,0,8,0, 0,0,0,0,0,0,4,8,2, 1,1, 2,
0,0,0,0,0,0,7,3,2,0, 0,0,0,0,0,0,3,8,8,9,1)
```

В данном представлении для задания адресов использовалось шестнадцатеричное представление и знаки «?» и «*» заменены на –1. Видно, что данная хромосома содержит 57 генов.

Это правило проверяется на архивных записях сетевых соединений. Если правило способно обнаружить аномальное поведение, то

Таблица 5.10. **Атрибуты, используемые ГА**

Атрибут	Диапазон значений	Пример значений	Описание
Адрес источника	0.0.0.0—255.255.255.255	0a.00.*.* (10.0.??.??)	Подсеть с адресами 10.0.0.0—10.0.255.255
Адрес назначения	0.0.0.0—255.255.255.255	c0.a8.b.*.* (192.168.10+??б??)	Подсеть с адресами 192.168.10.0—192.168.255.255
Порт источника	0—65535	33337	
Порт назначения	0—65535	00080	HTTP
Продолжительность соединения	0 — 999 999 999	000000482	482 с
Состояние соединения	1—20	11	Соединение закончено инициатором
Протокол	1—9	2	TCP
Число байт, посланное инициатором	0 — 999 999 999	0000007320	7 320 байт
Число байт, полученное приемником	0 — 999 999 999	0000038891	38 891 байт

Примечание. Знаки «?» и «*» обозначают любое значение, знак «+» означает добавление любого значения.

при получении новой популяции его хромосоме дается «поощрение». Если же это правило укажет на нормальное соединение, то его хромосома «наказывается».

Так как одно правило не может обнаружить все аномальные соединения, необходима популяция. Начальная популяция формируется случайным образом. Последовательно применяются операторы ГА. Применение функции оценивания позволяет наилучшим правилам смещаться к правилам, которые соответствуют аномальным соединениям.

Рассмотрим некоторые параметры ГА данного эксперимента (каждый из них сильно влияет на эффективность ГА).

Функция оценки определяет расхождение и соответствие. Расхождение вычисляется в зависимости от того, соответствует ли поле соединения множеству данных до классификации, и умножается на вес соответствующего поля (каждое поле соединения имеет различную важность):

$$\text{Расхождение} = \sum \text{соответствие} \times \text{вес},$$

где суммирование ведется по всем 57 генам.

Значения весов задаются в следующем порядке (по убыванию): адрес назначения, адрес источника, порт назначения, время соединения, количество посланных инициатором соединения байт, количество принятых приемником байт, состояние, протокол, порт источника (такое расположение полей соответствует порядку полей, фиксируемых анализаторами, сниферами). При соответствии значения поля правилу устанавливается значение 1, при несоответствии — 0).

Для системы устанавливается уровень подозрительности — порог, который показывает границу, до которой считается, что два различных соединения совпадают. Действительное значение уровня подозрительности определяется из наблюдаемых архивных данных. Вычисляется абсолютная разность между расхождением хромосомы и уровнем подозрительности:

$$\Delta = |\text{расхождение} - \text{уровень подозрительности}|.$$

Ранжирование показывает, легко ли идентифицируется аномалия. Через ранжирование определяется величина наказания:

$$\text{Наказание} = (\Delta \times \text{ранг})/100.$$

Это позволяет вычислить сходство хромосомы:

$$\text{Сходство} = 1 - \text{наказание}.$$

Видно, что значение сходства находится в интервале от 0 до 1.

Используя ГА, необходимо найти локальные максимумы (множество хороших решений) вместо глобального максимума (одного наилучшего решения), так как множество правил обнаружения лучше одного правила.

Для нахождения множества локальных правил можно использовать *технологию ниш*. Данная технология также заимствована из живой природы. Она основывается на том, что внутри каждого окружения есть различные подпространства (ниши), в которых могут существовать различные виды жизни. Подобно этому, ГА может устанавливать разброс каждой популяции в одном многомодальном домене, который соответствует доменам, требуемым для идентификации множества оптимумов.

При использовании технологии ниш применяются два основных подхода: группирование и разделение. При группировании используются наиболее сходные члены популяции для замены в новой популяции, чтобы замедлить ее движение к одной точке. При разделении уменьшают сходные элементы, что заставляет популяцию перемещаться к локальным максимумам, которые могут быть менее популятивны. При этом для оценки сходства могут использоваться различные метрики, например расстояние Хемминга между битами хромосом.

Особенностью применения оператора мутации для рассмотренной задачи обнаружения является возможность выхода поля (гена) хромосомы за разрешенные пределы, например выход значения адреса за значение 255. Это обстоятельство требует контроля результатов применения операторов. К числу других особенностей необходимо отнести следующие: число хромосом в начальной популяции, вероятности скрещивания и мутации, а также допустимое число поколений. Эти и другие особенности приводят к тому, что использование ГА для обнаружения вторжений в настоящий момент носит только исследовательский характер, хотя результаты показывают перспективность данного подхода.

5.5.5. Применение нейронных сетей

Нейронные сети (НС) — это составляющая часть искусственного интеллекта, в которой для обработки сигналов используются явления, аналогичные происходящим в нейронах живых существ. Важнейшая особенность сети, свидетельствующая о ее широких возможностях и огромном потенциале, состоит в параллельной обработке информации всеми звеньями сети. При громадном количестве межнейронных связей это позволяет значительно ускорить процесс обработки информации.

Другое не менее важное свойство — способность к обучению и обобщению накопленных знаний. Натренированная на ограниченном множестве данных сеть способна обобщать полученную информацию и показывать хорошие результаты на данных, не использовавшихся в процессе обучения.

Существуют различные способы объединения отдельных нейронов между собой и организации их взаимодействия, что привело к созданию сетей различных типов. Каждый тип сети, в свою очередь, тесно связан с соответствующим методом подбора весов межнейронных связей (т. е. обучения).

Искусственные нейронные сети возникли на основе знаний о функционировании нервной системы живых существ. Они представляют собой попытку использования процессов, происходящих в нервных системах, для выработки новых технологических решений.

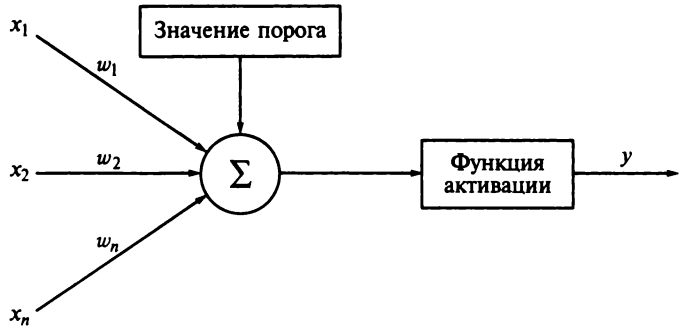


Рис. 5.10. Структурная схема модели нейрона МакКаллока-Питтса

Нервная клетка, сокращенно называемая нейроном, является основным элементом нервной системы. Как и у любой другой клетки, у нейрона имеется тело со стандартным набором органелл, называемое *сомой*, внутри которой располагается ядро. Из сомы нейрона выходят многочисленные отростки, играющие ключевую роль в его взаимодействии с другими нервными клетками. Можно выделить два типа отростков: многочисленные тонкие, часто ветвящиеся *дендриты* и более толстый, расщепляющийся на конце *аксон*.

Входные сигналы поступают в клетку через синапсы, тогда как выходной сигнал отводится аксоном через его многочисленные нервные окончания, называемые *колатералами*. Колатералы контактируют с сомой и дендритами других нейронов, образуя очередные синапсы. Передача сигналов внутри нервной клетки является сложным электрохимическим процессом.

В одной из первых моделей нейрона, называемой моделью МакКаллока—Питтса (предложенной в 1943 г.), нейрон считается бинарным элементом. Структурная схема этой модели представлена на рис. 5.10.

В данной модели входные сигналы x_j ($j = 1, 2, ..., N$) суммируются с учетом соответствующих весов w_{ij} (сигнал поступает в направлении от узла i к узлу j) в сумматоре Σ, после чего результат сравнивается с пороговым значением w_{i0}. Выходной сигнал нейрона y_i определяется при этом зависимостью

$$y_i = f(\sum w_{ij} x_j(t) + w_{i0}).$$

Функция $f(u_i)$ называется функцией активации. В модели МакКаллока — Питтса это пороговая функция вида

$$f(u) = \begin{cases} 1 \text{ для } u > 0; \\ 0 \text{ для } u \le 0. \end{cases}$$

Коэффициент w_{ij} представляет собой вес семантических связей. Положительное значение веса соответствует возбуждающим синап-

сам, отрицательное значение — тормозящим, тогда как $w_{ij} = 0$ свидетельствует об отсутствии связи между i-м и j-м нейронами.

Модель МакКаллока — Питтса — это дискретная модель, в которой состояние нейрона в момент $(t + 1)$ рассчитывается по значениям его входных сигналов в предыдущий момент t.

Через несколько лет Хебб в процессе исследований ассоциативной памяти предложил теорию обучения (подбора весов w_{ij}) нейронов. При этом он использовал наблюдение, что веса межнейронных соединений при активации нейронов могут возрастать. В модели Хебба приращение всех весов Δw_{ij} в процессе обучения пропорционально произведению выходных сигналов y_i и y_j нейронов, связанных весом w_{ij}:

$$w_{ij}(k + 1) = w_{ij}(k) + \eta y_i(k) y_j(k),$$

где k — номер цикла; η — коэффициент обучения.

В начале 60-х годов XX в. Видроу предложил теоретическое обоснование и сформулировал принципы практической реализации адаптивных устройств обработки сигналов, что стало существенным вкладом в развитие нейронных сетей. В 1962 г. была опубликована книга Розенблатта, в которой представлена теория динамических нейронных систем для моделирования мозговой деятельности, основанная на перцептронной модели нервной клетки. В этой теории использовалось представление нейрона моделью МакКаллока — Питтса, в которой функция активации принимала двоичное значение 0 и 1.

Ограниченные возможности одиночного перцептрона и составляемых из таких элементов одноуровневых сетей подверглись критике в книге М.Минского и С.Пейнерта, что вызвало резкое снижение финансирования этой сферы научных исследований и привело к замедлению развития искусственных нейронных сетей.

Очередному возрастанию интереса к сетям параллельной обработки информации, которыми считаются и нейронные сети, способствовало бурное развитие в 80-х годах XX в. технологии производства устройств сверхвысокой степени интеграции. Начиная с опубликованных в 1982 г. работ Хопфилда теория нейронных сетей развивается в стремительном темпе.

Рассмотрим некоторые подходы применения аппарата нейронных сетей для обнаружения вторжений. Первые попытки (в конце 90-х годов XX в.) применения технологии НС относились к хостовым СОВ. Основной упор в этих работах делался на то, что текущее поведение пользователя может быть предсказано на основании параметров, полученных в процессе обучения системы обнаружения (параметры и характеристики предыдущего поведения).

В настоящее время центр исследований в области применения НС к проблемам обнаружения вторжений сместился к сетевым СОВ, что объясняется следующими основными достоинствами нейронных сетей:

• способность к быстрой обработке данных вследствие параллельности самой структуры нейронной сети;

• способность к обучению и самоорганизации.

Сетевые СОВ на базе НС могут анализировать сетевой трафик в реальном масштабе времени, что в силу непрерывного увеличения пропускной способности каналов связи является одним из основных требований к СОВ.

В зависимости от типа используемых НС направления применения НС для построения СОВ можно классифицировать следующим образом:

1) многоуровневые нейронные сети прямого распространения (Multi Layers Feed-Forward, MLFF) — первые работы с такими сетями фокусировались на обнаружении аномалий поведения пользователя (например, рассмотрение перечня исполняемых пользователем команд);

2) рекуррентные и адаптивные НС — в таких системах производится анализ реакции защищаемой системы или сети на результат работы нейронной сети, что позволяет оценивать корреляцию текущих выходов нейронной сети с предыдущими входами и состояниями. Примером таких систем является ELMAN (Cerebellar Model Articulation Controller);

3) нейронные системы без учителя — большинство таких систем использует самоорганизующиеся шаблоны (Self-Organizing Maps, SOM) для обучения характеристикам нормальной системной активности и идентификации статистических отклонений от нормальных характеристик.

В качестве примеров использования технологии для построения СОВ рассмотрим два случая: использование BP-сетей (Back Propagation, BP) и сетей без учителя — SOM. Примером первого подхода является система обнаружения вторжений с BP-классификатором (рис. 5.11).

Основными компонентами такой СОВ являются монитор пакетов, модуль извлечения свойств, BP-классификатор (анализатор) и генератор сигналов тревоги.

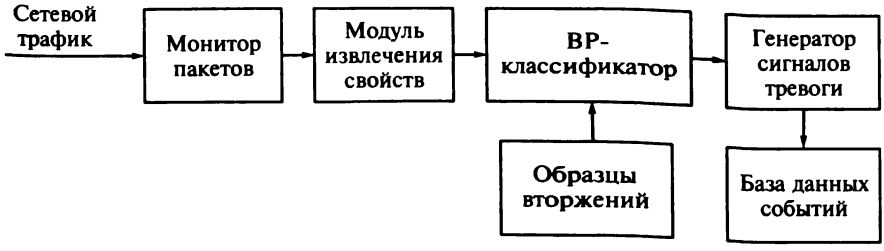


Рис. 5.11. Система обнаружения вторжений с BP-классификатором

Монитор пакетов представляет собой обычную сетевую карту, работающую в promiscuous режиме. Для реализации СОВ на платформе Windows может быть применена библиотека winpcap.

Модуль извлечения свойств (выбора) является важным элементом, поскольку сам по себе сетевой трафик не пригоден для входа в классификатор. Модуль извлечения свойств формирует вектор параметров, который представляет собой основные характеристики пакета. В качестве таких параметров в рассматриваемом прототипе выбраны: тип протокола, длина заголовка пакета, контрольная сумма пакета, номер порта, флаги TCP-протокола и т.д.

Структура вектора для описания атак имеет вид, приведенный ниже (расшифровка обозначений прямо следует из обозначений полей заголовков пакетов):

Attack(type)=(P-id, H-len, C-sum, S-port, D-port, Icmp-type, Icmp-code, Flag, P-len, P-data)

Примеры структур векторов конкретных атак приведены ниже (содержимое параметров векторов следует из структуры вектора, значением *null* обозначается отсутствие параметра):

Attack(CGI)= (TCP, 32, 0, 2345, 80, null, null, A, 421, get cgi-bin)

Attack(Redirect)= (ICMP, 20, null, null, null, 8, 3, null, 192, Ia)

Attack(FTP)= (TCP, 24, 16, 21, 21, null, null, PA, 256, ROOT)

Attack(UDP)= (UDP, 16, 10, 138, 126, null, null, null, 448, 3c)

Модель BP-классификатора является наиболее широко распространенной моделью НС, которая достаточно хорошо зарекомендовала себя для решения задач распознавания образов. Основные проблемы, рассматриваемые при разработке такого BP-классификатора, следующие:

- число скрытых уровней — согласно теории 3-уровневая модель BP-классификатора может представить соответствие размерности n к размерности m, поэтому для большинства приложений одного скрытого слоя достаточно;
- размерность входного слоя определяется числом выбранных параметров для анализа (двоичным представлением параметров), размерность выхода — числом решений, которое будет принимать классификатор;
- размерность скрытого слоя определить трудно — реальное число входов скрытого уровня определяется в результате экспериментов;
- выбор функции передачи (функции весов) определяется экспериментально, для BP-моделей можно использовать функцию $\text{Logsig}(x) = 1/(1 + \exp^{-x})$;

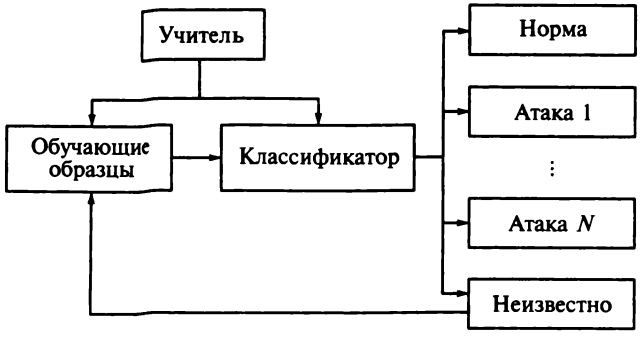


Рис. 5.12. Схема обучения классификатора

• инициализация весов и порогов производится случайными числами в заданном диапазоне ($w_{ij} \in [0; 1]$, $u_i \in [-0,5; 0,5]$).

Процесс формирования СОВ включает в себя три основных стадии: обучение на массивах тренировочных данных, тестирование правильности работы на других массивах данных и собственно обнаружение атак в ходе эксплуатации.

Схема обучения классификатора приведена на рис. 5.12.

В процессе обучения классификатор (анализатор) учится тому, что является нормальным, а что аномальным (причем анализатор указывает название атаки). После обучения классификатор запоминает полученные знания (все значения связей и порогов) и готов к эксплуатации. В процессе тестирования проверяется функционирование классификатора, подсчитываются частоты ложных тревог и пропусков. В случае необходимости может проводиться дополнительное обучение на новых массивах данных. Одним из вариантов решения анализатора является решение «неизвестно», т. е. анализатор не смог определить входные данные как нормальные и в то же время не смог отнести их ни к одной из известных ему атак. Появление такого решения может свидетельствовать о наличии новой атаки, обучение на которой не проводилось. После анализа соответствующих данных вектор такой атаки могут быть вставлены в тренировочные данные, чтобы классификатор далее обнаруживал атаки данного типа. Такой подход позволяет динамически изменять возможности классификатора, а значит, и СОВ в целом.

Генератор сигналов тревог данной модели кроме выдачи сигнала администратору о типе обнаруженной атаки проводит запись аудита, в которой указываются дата, время, источник, тип атаки, степень опасности (извлекаемая по типу атаки из БД) и другая информация. Проведенные авторами эксперименты показали очень хорошие результаты при обнаружении атак отказа в обслуживании, Land, Ping of Death, Smurf, CGI, Fingerprint, а также при обнаружении других неизвестных анализатору атак отказа в обслуживании.

Рис. 5.13. Схема построения системы обучения без учителя

В качестве примера использования НС без учителя рассмотрим систему UNNID (Unsupervised Neural Net based Intrusion Detection). Схема построения системы приведена на рис. 5.13.

Предлагаемая система включает в себя модуль предоставления данных (который может функционировать как в режиме реального времени, так и в отложенном режиме), препроцессор, анализатор, модуль оценивания и модуль генерации ответа.

Модуль представления данных может обрабатывать файлы аудита или получать данные из сетевых пакетов.

Препроцессор преобразует текстовые данные в цифровые и, при необходимости, в двоичный или нормализованный вид.

Анализатор использует полученные данные для обучения и тестирования (потом для анализа и обнаружения атак в реальном масштабе времени или отложенном). Выходной сигнал (норма или тип атаки) передается модулю генерации сигнала тревоги.

Модуль оценивания формирует отчеты о работе системы и определяет показатели качества работы системы.

Модуль генерации ответа осуществляет запись аудита, соответствующую обнаруженной атаке.

5.5.6. Языки описания атак

В настоящее время исследователями и разработчиками СОВ предложено и разработано множество языков, предназначенных для описания вторжений и их обнаружения. Среди них особое место занимают языки STATL, SISL, P-BEST, CYCL и NASL.

STATL — один из наиболее хорошо определенных языков, предназначенных для распознавания атак. Сила этого языка в использовании сходных с языками программирования конструкций для описания последовательных, условных и итеративных событий. Эти конструкции прямо адресуются к структурам многих автоматических атак. Хотя язык является расширяемым, в нем не используются средства для представления переменных времен и длительностей, так как STATL не снабжен абстрактными временами или временными интервалами. Он оперирует только с конкретными временами и интервалами, что вызывает проблемы при описании сценариев атак.

SISL — представляет значительный шаг вперед в создании языка отчетов для множества СОВ. Его разработка как языка отчетов не предусматривает возможность использования для распознавания атак. Однако SISL, используемый совместно с STATL, фокусируется на представлении конкретных времен и интервалов. Расширения языка связаны с распознаванием скелетных конструкций для формирования отчетов.

P-BEST — архитектура СОВ Emerald идет дальше в направлении от простых сигнатур и аномалий, предоставляя возможность использования корреляции данных сенсоров и процессов. Язык P-BEST содержит формализмы и шаблоны для реализации вероятностных и лингвистических правил, что позволяет распознавать различные события. Для каждого шаблона можно разработать множество P-BEST правил, которые охватывают всевозможные проверки соответствий данным.

CYCL — предназначен для представления знаний. Он предоставляет значительные мощные конструкции для описания абстрактных и конкретных концепций между различными доменами, включая логические описания, немонотонные описания, модульные и внутренние конструкции, а также вероятностные конструкции. Стандартная онтология, предоставляемая CYCL, содержит концепции временных точек и интервалов, процессов и т. д. Сам язык является достаточно абстрактным, так как не предназначен для распознавания событий.

NASL — входит в состав сканера уязвимостей Nessus, который предназначен для автоматического поиска известных изъянов в защите информационных систем. Он способен обнаружить наиболее часто встречающиеся уязвимости, включая наличие уязвимых версий служб, наличие открытых портов, ошибки конфигурирования и наличие слабых паролей. Проверка наличия уязвимостей осуществляется по базе данных уязвимостей, которая пополняется еженедельно. Кроме того, сканер содержит специальный язык сценариев NASL (Nessus Attack Scripting Language), позволяющий разрабатывать пользовательские скрипты для проверки определенной уязвимости. С помощью языка NASL можно сконструировать и отправлять различные сетевые пакеты с произвольным содержимым. Разработчики Nessus гарантируют, что NASL-сценарий не пошлет пакеты

ни на какой хост, кроме выбранной цели, и не будет выполнять на локальном компьютере никаких команд. Цель NASL — тестирование безопасности сетевых сервисов. Написание модулей на языке NASL достаточно просто, благодаря переносимому и легко читаемому коду, а также наличию общих черт с языком C.

5.6. Другие методы обнаружения вторжений

Кроме рассмотренных методов обнаружения нарушений существуют дополнительные методы, выполняющие обнаружение. Все системы обнаружения вторжений можно классифицировать по подходам к обнаружению атак и вторжений.

1. *Предварительный этап — подготовка к обнаружению вторжений.* На этом этапе анализируется состояние защищаемой системы, для чего используются специальные средства — системы анализа защищенности (security assessment systems) или сканеры безопасности (security scanners).

2. *Собственно этап обнаружения.* На этом этапе функционируют собственно системы, позволяющие выявлять атаки в процессе их реализации. Именно эти средства принято называть системами обнаружения атак и вторжений. На этом этапе функционируют и специально создаваемые для взлома, так называемые обманные системы (deception systems), а также системы Honeynet.

3. *Этап обнаружения уже совершенных атак*, т.е. обнаружение следов вторжений. Такие системы, в свою очередь, можно разделить на два класса: системы контроля целостности (integrity checkers), отслеживающие изменения контролируемых ресурсов, и системы анализа журналов регистрации (log checkers).

Данная классификация приведена на рис. 5.14.

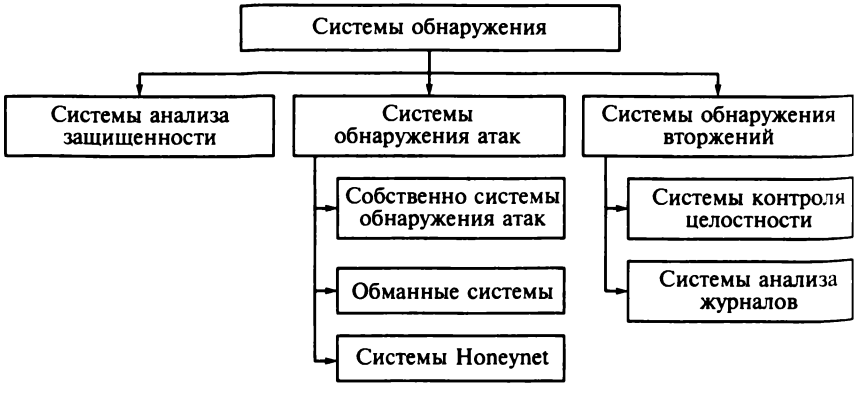


Рис. 5.14. Классификация систем обнаружения

5.6.1. Системы анализа защищенности

Системы анализа защищенности (security assessment systems), также называемые сканерами безопасности (security scanners) или системами поиска уязвимостей, проводят всесторонние исследования контролируемых ресурсов с целью обнаружения уязвимостей, которые могут привести к нарушениям политики безопасности, т.е. к реализации атак. Результаты, которые получаются от средств анализа защищенности, представляют собой «мгновенный снимок» (snapshot) состояния защищаемой системы на данный момент времени. Главная цель таких систем — определить слабости защищаемой системы, т.е. потенциальную возможность реализации атак.

Системы анализа защищенности можно разбить на три класса по типам обнаруживаемых ими уязвимостей: системы определения уязвимостей проектирования, системы поиска уязвимостей реализации и системы поиска уязвимостей конфигурации (эксплуатации).

Системы анализа защищенности могут функционировать на всех уровнях информационной структуры организации, т.е. на уровне сети и уровне хоста. Анализ защищенности хоста может быть направлен на анализ ОС, системы управления базой данных или прикладного ПО.

При проведении такого анализа используются две основные стратегии: пассивная и активная.

Пассивная стратегия заключается в анализе файлов конфигурации, системного реестра, имеющихся паролей, а также других системных объектов. Примером реализации подобной стратегии является средство MBSA (Microsoft Baseline Security Analyzer). В настоящее время компания Microsoft предоставляет версию 2.2 (http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=7558), которая поддерживает практически все последние версии ОС Windows, включая 64-битные платформы. Данное средство позволяет анализировать отдельный компьютер и множество компьютеров локальной сети.

Анализу подвергаются следующие вопросы: нахождение административных и SQL-уязвимостей, наличие слабых паролей, определение административных IIS (Internet Information Services) и SQL-уязвимостей, проверка наличия последних обновлений.

По результатам анализа выдается отчет, содержащий следующие разделы:

1. Наличие загруженных дополнений.
2. Административные уязвимости, среди которых время жизни паролей, включение (отключение) Windows МЭ, слабость имеющихся паролей, наличие учетных записей guest, ограничения анонимного доступа и количество административных учетных записей.
3. Административные уязвимости для IIS (включение аудита, наличие служб, наличие совместно используемых файлов).

4. Административные SQL-уязвимости (учетные записи службы, установленная политика паролей и разрешений доступа к папкам, количество имеющихся sysadmins, режим аутентификации SQL-сервера, наличие учетной записи guest и установленные разрешения на доступ к реестру).

5. Административные уязвимости приложения — установленная зона Internet Explorer.

Каждое сообщение отчета сопровождается ссылкой на подсказку для устранения найденной уязвимости.

Активная стратегия заключается в проведении реальных атак на защищаемую систему и анализе полученных результатов.

Одним из свободно распространяемых средств, принадлежащих к активной стратегии, является сетевой сканер Nmap (www.insecure.org). Этот сканер предназначен для сканирования различных сетей, определения состояния сканируемых сетей, открытых портов и соответствующих им служб. Кроме того, Nmap позволяет определить ОС удаленного хоста с помощью множества методов сканирования. База данных отпечатков (fingerprints) всех существующих ОС периодически пополняется.

Сканер содержит множество опций, которые позволяют определить, например, следующие характеристики сканируемого хоста: метод генерации TCP ISN, имя пользователя (username) — владельца процесса, который зарезервировал сканируемый порт, и символьные имена, соответствующие сканируемым IP-адресам.

Системы анализа приложений стали появляться недавно. Возможности по анализу защищенности приложений реализованы как в системах анализа сетевых сервисов, так и в системах анализа ОС. Однако возможности эти не полны и реализованы только для широко распространенного прикладного ПО.

Системы анализа защищенности могут применяться специалистами по защите информации для оценки уровня защищенности и контроля правильности (соблюдения правил установленной политики безопасности) настроек сетевого, системного и прикладного ПО. Кроме того, системы анализа защищенности применяются при проведении аудита безопасности организации.

5.6.2. Системы анализа целостности

Системы анализа целостности позволяют определить факт вторжения, т. е. успешной атаки на защищаемую систему. Если злоумышленник проник в защищаемую систему, то, как правило, он устанавливает на пораженной системе потайной ход или другие вредоносные программы, применяет методы сокрытия факта вторжения или пытается блокировать функционирующую систему защиты. Это связано главным образом с изменением каких-либо файлов (испол-

няемых, конфигурационных, динамических библиотек, драйверов и т. п.) пораженной системы.

Системы контроля целостности используют криптографические проверки основных защищаемых файлов, которые позволяют определить факт модификации этих файлов. Чаще всего для этой цели используются хеш-функции, которые даже при незначительных изменениях в контролируемом файле формируют различные хеш-значения.

Алгоритмы вычисления хэш-функций являются криптографически стойкими, т. е. при заданном конкретном входном значении практически невозможно получить идентичный результат для любого другого входного значения.

Функционирование систем контроля целостности включает в себя два этапа. На *первом* (*предварительном*) *этапе* осуществляется: вычисление контрольных значений защищаемых файлов или других объектов (хеш-значения, длины файлов, время их создания и т. п.) и сохранение их в определенной области. На *втором этапе* проводится периодический контроль защищаемых файлов с целью обнаружения изменений. Таким образом, системы контроля целостности определяют только факт изменения защищаемых файлов, т. е. являются пассивными системами обнаружения вторжений. Обычно при обнаружении подобных изменений в защищаемых файлах системы контроля целостности генерируют сигнал тревоги. Существуют системы, которые не только обнаруживают факт изменения защищаемых файлов, но и восстанавливают их сохраненное состояние. Примером подобной системы является система защиты файлов Windows (Windows File Protection, WFP).

Механизм WFP был введен для ОС Windows XP и отвечает за защиту важных системных файлов, устанавливаемых вместе с ОС (например, файлы с расширениями .dll, .exe, .ocx и .sys, а также некоторые шрифты True Type). Суть механизма состоит том, что в системном процессе winlogon.exe работает следящий поток. Если он замечает изменение системного компонента, то компонента восстанавливается из копии, сохраненной в специальном кеше. Защищаемые системные файлы и директории подписываются и сохраняются в специальной директории (папке), заданной по умолчанию (%systemroot%\system32\dllcache). Местоположение данной папки может быть изменено администратором. Замена таких защищаемых файлов (около 1 700) возможна только при выполнении следующих операций:

- установка пакетов обновления для Windows с помощью программы Update.exe;
- установка исправлений с помощью программ Hotfix.exe и Update.exe;
- обновление ОС с помощью программы Winnt32.exe;
- использование Web-узла Windows Update.

В качестве дополнительного механизма защиты, обеспечиваемого WFP, используется средство проверки системных файлов (Sfc.exe). Это средство проверяет, не были ли защищенные файлы изменены программами, установленными автоматически. Кроме того, программа Sfc.exe проверяет все файлы каталога, используемые для отслеживания правильных версий файлов. В случае отсутствия или повреждения любого из файлов каталога он переименовывается и восстанавливается из папки кеша. Если в папке кеша файл найти не удается, то WFP запрашивает установку соответствующего носителя с новой копией файла каталога. Данное средство предоставляет администратору возможность проверить версии всех защищенных файлов, а также проверяет и повторно заполняет папку кеша. Если папка кеша повреждена или стала непригодной для использования, то для восстановления ее содержимого используется команда sfc /scanonce или sfc /scanboot.

При переходе к ОС Windows Vista, Windows Server 2008 и Windows 7 компания Microsoft сформировала новую систему защиты — защита ресурсов (Windows Resource Protection, WRP), заменившую WFP. Данная система защищает от перезаписи важные системные файлы, папки и ключи реестра, инсталлированные как часть ОС.

Приложения не должны изменять защищаемые WRP ресурсы, поскольку они используются ОС и ее приложениями.

Механизм WRP защищает:

- важные для системы файлы с расширениями: .acm, .ade, .adp, .app, .asa, .asp, .aspx, .ax, .bas, .bat, .bin, .cer, .chm, .clb, .cmd, .cnt, .cnv, .com, .cpl, .cpx, .crt, .csh, .dll, .drv, .dtd, .exe, .fxp, .grp, .hls, .hlp, .hta, .ime, .inf, .ins, .isp, .its, .js, .jse, .ksh, .lnk, .mad, .maf, .mag, .mam, .man, .maq, .mar, .mas, .mat, .mau, .mav, .maw, .mda, .mdb, .mde, .mdt, .mdw, .mdz, .msc, .msi, .msp, .mst, .mui, .nls, .ocx, .ops, .pal, .pcd, .pif, .prf, .prg, .pst, .reg, .scf, .scr, .sct, .shb, .shs, .sys, .tlb, .tsp, .url, .vb, .vbe, .vbs, .vsmacros, .vss, .vst, .vsw, .ws, .wsc, .wsf, .wsh, .xsd и .xs;
- критические для работы директории. Механизм WRP защищает некоторые важные каталоги. Папка, содержащая только защищенные WRP-файлы, может быть заблокирована таким образом, что создание в ней файлов или подкаталогов разрешено лишь доверенному процессу;
- важные ключи реестра (а также подключи и их значения). Приложения могут использовать функции SfcIsFileProtected и SfcIsKeyProtected для определения факта, что данный файл или ключ реестра защищен;
- механизм WRP копирует файлы, необходимые для загрузки ОС в специальной директории (%Windir%\winsxs\Backup), а не все защищаемые файлы в механизме WFP. Размер этой директории и список защищаемых файлов не может быть изменен.

Механизм WRP работает, устанавливая дискретные списки доступа DACLs и ACL для защищаемых объектов. Разрешение на чтение-

Запись защищенных WRP-объектов допускается лишь процессам, использующим службу Windows Modules Installer (TrustedInstaller.exe). Теперь даже у администраторов нет прав полного доступа к системным файлам, т.е. полный доступ для системных файлов и ключей реестра имеет только системный пользователь TrustedInstaller. И при попытке изменить системный файл ОС выдает сообщение: «Запросите разрешение от TrustedInstaller на изменение этого файла».

Защита срабатывает, когда WRP получает сигнал об изменении защищаемого файла (директории, ключа реестра). Механизм WRP сверяет цифровую подпись файла в сохраненном каталоге для определения того, является ли измененный файл корректной версией. Если файл не соответствует подписи, он заменяется правильной версией из кеша или источника инсталляции ОС. Поиск корректных версий осуществляется в следующем порядке:

1) поиск в директории кеша;

2) поиск сетевого пути инсталляции, если система инсталлировалась по сети;

3) поиск Windows CD-ROM, если система инсталлировалась с CD-ROM.

Это позволяет обеспечить защиту основных системных файлов и параметров реестра от модификации и удаления.

5.6.3. Вспомогательные средства обнаружения

К числу вспомогательных средств обнаружения, которые не попадают в приведенную классификацию, но имеют важное значение для обнаружения, относятся средства контроля целостности, обманные системы (padding cells) и хосты-приманки (Honeypot).

Средства контроля целостности (как правило, файлов) вычисляют образы защищаемых файлов и сохраняют их в определенном месте. В качестве образов обычно используют значения хэш-функций. Периодически средство контроля целостности проводит вычисление образов и сравнение их с хранящимися. При несовпадении значений образов генерируется сигнал тревоги.

Обманные системы располагаются в специальном месте защищаемой сети. Когда СОВ обнаруживает атаку, трафик атакующего переключается на обманную систему, в которой он может делать все, что заблагорассудится. Обманная система функционирует в моделируемом окружении, поэтому атакующий не может нанести вреда защищаемой сети. Обычно моделируемое окружение обманной сети заполняется интересными данными, чтобы атакующий убедился, что его атака идет по плану. Сама обманная система предоставляет возможность контроля действий нарушителя. Существуют различные варианты обманных систем, среди которых можно выделить серверы пользовательского уровня. В этом случае на хосте, служащем

приманкой (honeypot), разворачиваются серверы пользовательского уровня (каждый сервер обладает полнофункциональной операционной системой и выполняется как пользовательский процесс).

Серверы помещаются в прикладное пространство хостовой ОС, поэтому каждому из них может быть назначен свой IP-адрес. Данная система моделирует целую локальную сеть, где каждый сервер пользовательского уровня представляется отдельным хостом этой сети (рис. 5.15).

Под honeypot в современном компьютерном мире понимается система, специально разработанная для того, чтобы на нее напали. Главной ее задачей является регистрация и контроль всех действий нарушителя.

Впервые эта идея рассмотрена в статьях Стола (Cliff Stoll), Белловина (Steve Bellovin) и Чезвика (Bill Chezwick). В этих статьях описывался анализ действий взломщиков компьютерных систем на основе подробных записей об их действиях. Термин «Honeypot» появился

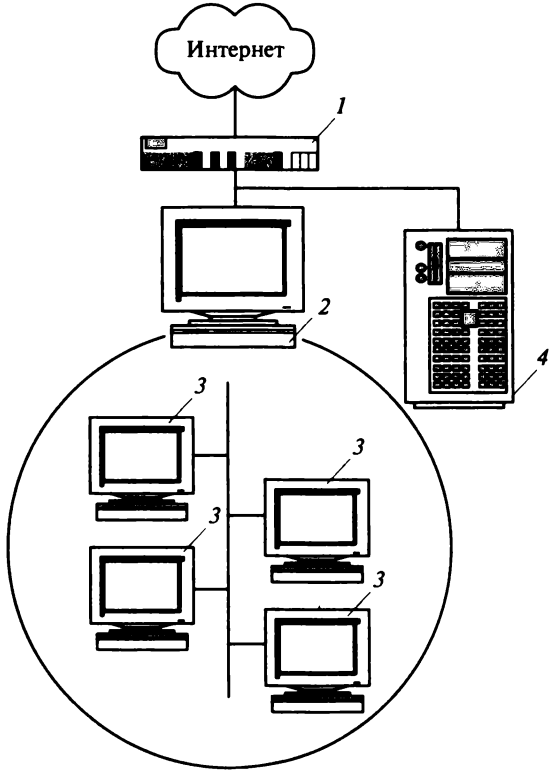


Рис. 5.15. Хост-приманка с серверами пользовательского уровня:
1 — маршрутизатор; *2* — хост Honeynet; *3* — сервер; *4* — сервер регистрации

позднее, но под ним подразумевали то же самое: установка систем, которые покажутся привлекательными для сетевых взломщиков и смогут проводить мониторинг всех действий, происходящих во взломанной системе. Анализ результатов такого мониторинга позволит узнать методы, средства и уязвимости, используемые взломщиками.

Традиционно Honeypot представляет собой отдельную систему (отдельный компьютер), соединенную с внешней сетью (Интернетом). В настоящее время существует несколько продуктов и средств, позволяющих создавать собственные honeypot, например:

- Deception Toolkit (http://www.all.net/dtk/);
- Cybercop Sting (http://www.pgp.com/products/cybercop-sting/);
- Resource Mantrap (http://www.resource.com/products/mantrap/).

Анализ данных подходов и продуктов, а также необходимость обеспечения защиты пользователей других систем от атак со стороны взломанных Honeypot привели Л.Шпицнера к разработке и созданию проекта Honeynet. В состав участников проекта входят известные профессионалы в области компьютерной безопасности, среди них Дэвид Дитрих, Эд Скаудис, Мартин Рош, Макс Вижн, Офир Аркин, Стюарт МакКлур и др.

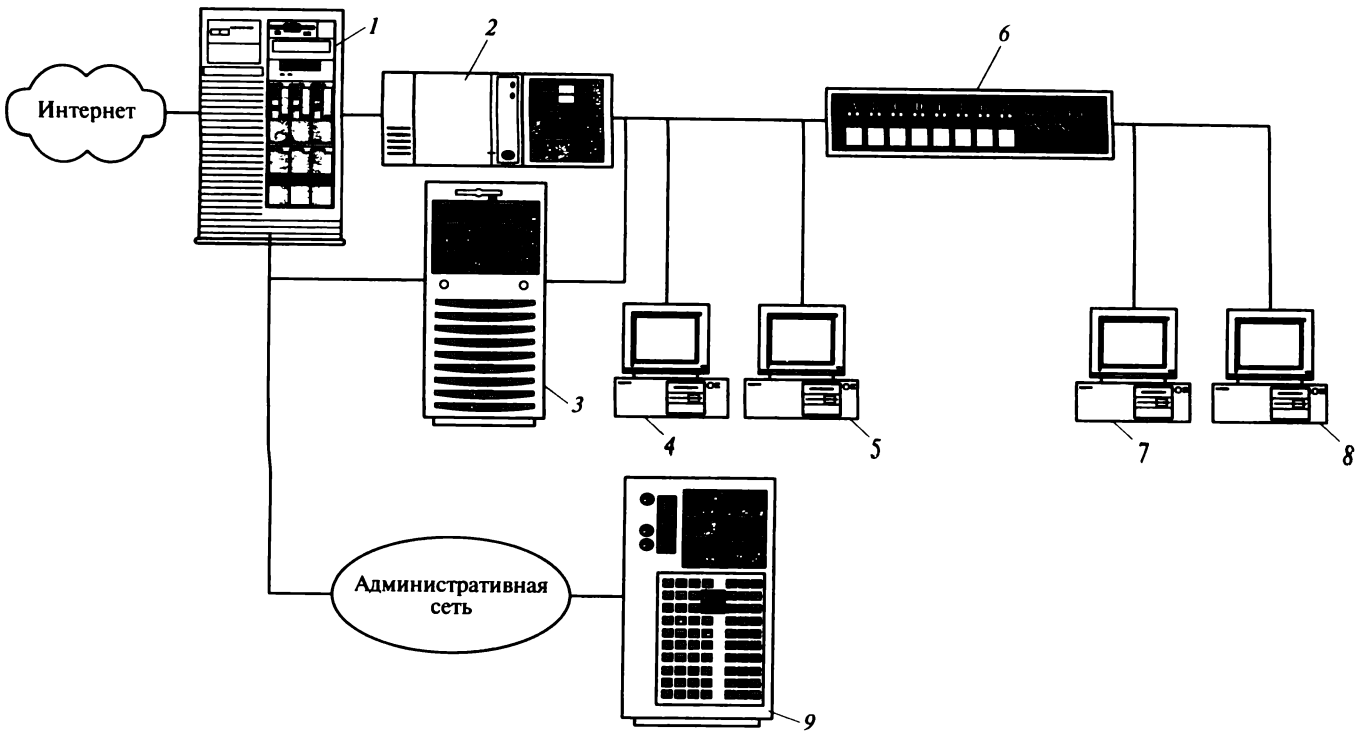
Honeynet представляет собой сеть, созданную специально для взлома. Она записывает и контролирует все входные и выходные данные. В пределах проекта Honeynet размещаются различные ОС, различные сетевые устройства и т.д. Все системы, находящиеся в проекте Honeynet, являются реальными стандартными системами и приложениями. Схема сети, используемая в экспериментах проекта Honeynet, приведена на рис. 5.16.

Основная цель создания Honeynet заключается в сборе максимально возможного количества информации о действиях злоумышленника. Анализ полученных данных не только позволяет установить неизвестные слабые места различных реальных систем, но и выработать рекомендации по защите аналогичных систем. Одно из преимуществ Honeynet — эта сеть не только знакомит нас с сообществом хакеров, но и позволяет определить наши собственные возможности в области обеспечения безопасности.

Honeynet — это механизм изучения инструментов, тактики и мотивов сообщества взломщиков. Эта система уникальна тем, что ничего не имитируется. Создается полностью контролируемая сеть из машин с ОС и приложениями, которые идентичны тем, которые используются реальными пользователями и организациями. После того как системы взломаны, они помогают не только понять действия взломщиков, но и определить риски и слабости, существующие в реальных системах.

При эксплуатации системы Honeynet (а также Honeypot) необходимо учитывать следующие важные обстоятельства:

- в системе должна осуществляться запись данных — фиксация всех действий, происходящих в пределах Honeynet, в том числе на


Интернет
1
2
6
3
4
5
7
8
Административная сеть
9

уровне сети и составляющих ее систем. В системе проекта Honeynet используются три уровня записи данных: системные журналы защитных устройств (МЭ, СОВ), сетевые журналы и записи аудита систем, входящих в состав Honeynet;

- поскольку система взламывается, необходимо обеспечить такую защиту проводимых записей, которую нарушитель не сможет изменить или уничтожить;
- скрытность проведения операций контроля и записи — нарушитель не должен иметь возможности обнаружить, что его действия контролируются и фиксируются. Реальные записи аудита систем должны вестись в обычном режиме;
- системы Honeynet нуждаются в постоянном администрировании и контроле в целях максимального уменьшения риска нанесения ущерба другим пользователям.

Еще одной целью проекта Honeynet является рост количества подобных сетей и развертывание их во всем мире. В этом случае появляется возможность сопоставления данных, собранных в различных сетях, возможность определения общих тенденций и предсказания будущих нападений.

5.7. Методы обхода систем обнаружения вторжений

Системы обнаружения вторжений имеют свои слабые стороны и уязвимости, что может быть использовано нарушителем. Рассмотрим основные точки уязвимостей СОВ и методы, которые могут использовать нарушители для обмана СОВ.

Каждый из компонентов СОВ, указанных в модели CIDF, имеет свое уникальное назначение и может быть атакован по разным причинам.

Например, e-box, работающие с входными «сырыми» данными, действуют как «глаза и уши» СОВ. Атаки против элементов генерации событий позволяют блокировать реальные события, происходящие в контролируемой системе. Тогда атака против e-блоков сетевой СОВ может сделать недоступным получение пакетов из сети или сделать недоступным соответствующее декодирование этих пакетов.

Некоторые СОВ используют сложный (изощренный) анализ. В таких системах надежность используемого a-box очень важна, поскольку атакующий может обойти систему обнаружения. Кроме того, сложная техника обнаружения может предоставить различные пути для проведения атаки. С другой стороны, простейшие системы могут пропустить атаки, в которых атакующий маскирует свою деятельность сложным скоординированным взаимодействием или взаимосвязями.

Необходимость надежной базы данных очевидна. Атакующий, который может разрушить компоненты d-box, может защититься от

записи деталей атаки. Неправильно используемая БД может позволить атакующему замену или удаление зарегистрированных данных об атаке.

Подробное исследование, посвященное методам обмана и обхода СОВ, было проведено Птасеком (Thomas H. Ptacek) и Ньюшэмом (Timothy N. Newsham) в 1998 г. Основной слабостью сетевых СОВ является то, что они принимают решение на основе анализа сетевого трафика, поэтому не могут предсказать, как поведет себя целевая система при получении этого трафика. Кроме того, сами СОВ могут быть подвержены атакам отказа в обслуживании (выведение их из строя или исчерпание их ресурсов). Рассмотрим основные подходы к обходу сетевых СОВ.

Обход (обман) СОВ существенно зависит от того, достаточно ли сильна сама СОВ, чтобы СОВ и целевая система рассматривали разные потоки данных одинаковым образом. Рассмотрим два основных варианта.

Если СОВ слабее целевой системы, на которую направлена атака, то при принятии решения СОВ может рассматривать пакет, который целевая система рассматривать не будет. Атакующий может использовать эту возможность вставкой дополнительных пакетов, которые замаскируют атаку для СОВ. Поскольку эти дополнительные (специально сформированные) пакеты будут отброшены целевой системой, атака не будет обнаружена. Например, если СОВ для атаки phf имеет сигнатуру вида GET/phf, то атакующий может вставить дополнительную информацию, которая замаскирует реальную сигнатуру атаки, например GET/cgi-bin/phf?, или гораздо более длинную строку (помещаемую в пакеты, которые целевая система удалит) вида: GET/cgi-bin/pleasedontdetectthisforme? В этой строке вставлены элементы leasedontdetectt, is, orme, после отбрасывания которых целевая система получит пакет, содержащий phf.

Если СОВ сильнее целевой системы, то целевая система будет принимать пакеты, которые СОВ не будет рассматривать, что может быть достигнуто использованием метода вставки. Например, для атаки phf атакующий может вставлять элементы данной атаки в пакеты для СОВ с дополнительной информацией. Тогда СОВ отбросит эти пакеты, но они будут рассмотрены целевой системой. Такой подход может позволить замаскировать атаку для СОВ.

Другим общим методом обмана СОВ является метод фрагментирования. В этом случае используется различие способностей СОВ и целевой системы по проведению реассемблирования пакетов. Атакующий может сконструировать последовательность пакетов, которая будет скрывать сигнатуру проводимой атаки, используя знания о силе или слабости СОВ по сравнению с целевой системой.

При применении метода вставки нарушитель разрушает работу реассемблирования потока данных добавлением пакетов. Кроме того, вставленные пакеты могут перекрывать данные, содержащиеся

в исходных фрагментах атаки. При использовании метода обхода нарушитель разрушает реассемблирование потока данных так, чтобы СОВ не рассматривала часть этого потока.

Особенно трудно защититься от атак отказа в обслуживании, направленных против самой СОВ. Это также может быть достигнуто использованием метода вставки большого числа пакетов, чтобы вызвать пропуск отдельных пакетов СОВ, или применением многочисленной фрагментации.

5.7.1. Методы обхода сетевых систем обнаружения вторжений

Сетевые СОВ функционируют как пассивные устройства или невмешивающиеся мониторы сетевого трафика. Сетевые СОВ могут обнаруживать аномалии и злоупотребления (сигнатуры). Методы обнаружения аномалий для сетевых СОВ используются редко из-за необходимости большого периода времени для построения «нормального» поведения и большого числа ложных срабатываний. Обход сетевых СОВ обнаружения аномалий больше игра, чем умение (злонамеренный трафик не может превысить пороги аномальности, установленные в контролируемой сети). Поэтому далее рассмотрим СОВ обнаружения злоупотреблений. Основные методы обхода таких СОВ:

- сбивание с толку;
- фрагментация;
- шифрование;
- перегрузка.

Сбивание с толку — это процесс манипулирования данными таким образом, чтобы сигнатура СОВ не соответствовала проходящему пакету, который бы интерпретировался приемной стороной (например, посылка пакета, использующего кодирование, или добавление вспомогательных символов). В качестве примера рассмотрим строку ../../c:\winnt\system32\netstat.exe, которую не всякая СОВ интерпретирует в таком виде из строки %2e%2e%2f%2e%2e%2fc:\winnt\system32\netstat.exe. Однако Web-сервер, которому будет направлена данная команда, интерпретирует обе строки одинаково (так как они удовлетворяют правилам HTTP).

Фрагментация — это разбивка пакета данных на фрагменты, которые можно послать в различном порядке (и с различными временными интервалами между ними), что может обмануть СОВ, которая не производит реассемблирования.

Если СОВ проводит реассемблирование, то нарушитель может превысить физический объем памяти СОВ, отведенный для реассемблирования, путем посылки большого числа фрагментов, содержащих «мусор», или превысить временной промежуток, в течение ко-

торого пакет должен реассемблироваться. Например, если система обнаружения имеет сигнатуру, содержащую символы D ← A ← T ← ← A, которые должны поступать в последовательности, указанной стрелками, а нарушитель посылает их следующим образом: A ← A ← ← T ← (временная пауза) ← D, то СОВ может удалить первые три символа из буфера, не дождавшись прихода четвертого. Все эти символы будут реассемблированы на приемной стороне.

Шифрование — сетевая СОВ должна иметь возможность исследовать полезную нагрузку пакета, чему может противодействовать нарушитель, шифруя трафик. Для этого используются шифрованные SSL, SSH и IPSec туннели. Это позволяет нарушителю использовать средства безопасности целевого хоста против него же. Это может быть опасно, в случае когда система имеет одну и ту же корневую директорию для не зашифрованных (HTTP) и для зашифрованных (HTTPS) Web-страниц (по умолчанию это устанавливается в Microsoft IIS после инсталляции сертификата). В этом случае нарушитель может использовать любую атаку против Web-сайта с HTTPS, такую как SQL-вставка, переполнение буфера или обход (просмотр) директории. Поскольку HTTPS использует SSL, трафик шифруется, что позволяет ему проходить СОВ. Данная проблема усугубляется широким применением SSL VPN. Это, в свою очередь, создает следующие проблемы для СОВ. Во-первых, нарушитель может атаковать сеть, а СОВ не обнаружит зашифрованную атаку (даже если она обнаружит, то нарушитель не будет пойман из-за анонимности). Во-вторых, если СОВ может работать с зашифрованным трафиком, то нарушитель может установить большое число сеансов работы с множеством хостов, что потенциально может помешать сетевой СОВ расшифровать в реальном времени трафик, принадлежащий атаке.

Перегрузка (отказ в обслуживании) — этот метод переполнения сетевой СОВ может быть достигнут различными путями. Наряду с применением своей атаки производится «затопление» другими атаками с фиктивными адресами источников, что приведет к генерации громадного количества сигналов тревоги и помешает определить истинного нарушителя. Возможно «затопление» сетевой СОВ трафиком таким образом, чтобы СОВ оказалась не в состоянии анализировать каждый пакет.

5.7.2. Методы обхода хостовых систем обнаружения вторжений

Программное обеспечение хостовой СОВ может функционировать на различных уровнях. Некоторые системы контролируют целостность важных файлов, другие контролируют сетевые соединения, входные строки и системную память на наличие сигнатур.

Системы контроля файлов, такие как Tripwire, могут, в свою очередь, создать проблемы. Контроль каждого файла может привести к появлению значительного числа ложных тревог, кроме того, не все атаки модифицируют файлы. Для хостовых СОВ обычно используется комбинация обнаружения аномалий и обнаружения сигнатур. Одним из основных для хостовых СОВ является вопрос: если хост будет скомпрометирован, то как удержать нарушителя от манипулирования с элементами СОВ для предотвращения обнаружения атаки? Основными методами для этого являются:

- контроль расположения и целостности файлов;
- сбивание с толку;
- вставка нулевого знака в запрос после указания метода;
- перехват приложения.

Контроль расположения и целостности файлов — большинство известных методов обхода хостовых СОВ работают против систем обнаружения сигнатур. Большое число СОВ, использующих контроль файлов, используют алгоритм MD5 или его варианты. Теоретически два разных файла могут иметь одно и то же хеш-значение, но подбор соответствующего файла требует громадных затрат времени и ресурсов. Кроме того, база данных хеш-значений обычно защищена паролем или зашифрована отдельным ключом. В этом случае атакующий может использовать слабости в методе контроля файлов. Для большинства систем контроля указываются директории, файлы в которых не проверяются. Поэтому такие директории могут использоваться для обхода систем обнаружения. Когда нарушитель скомпрометирует систему, находясь в такой временной директории, он может удалить систему контроля файлов или пересчитать значения хеш-функции для измененных файлов в других директориях.

Атакующий может заменить программы, которые загружаются при старте системы, так как большинство хостовых СОВ контролируют файлы и скрипты загрузки.

Основной проблемой при контроле файлов является то, что система генерирует тревогу, когда вторжение уже произошло.

Сбивание с толку — это достаточно широко используемый метод, поскольку он работает и против сетевых, и против хостовых СОВ.

Рассмотрим пример манипуляции путей, которые соответствуют сигнатуре (табл. 5.12):

Приведенные примеры показывают одно и то же физическое расположение файла. Код ./ указывает на текущую директорию, поэтому можно добавлять любое их число. Код /// обычно интерпретируется приложением как один слеш, поскольку имя директории не может быть NULL. Аннулирование обхода более сложно, так как позволяет перемещаться по дереву директории и при использовании ../ команды перехода в предыдущую директорию позволяет вернуться в

Таблица 5.12. **Примеры манипуляции путей**

Манипуляция	Пример
Самоссылающиеся директории	*/_vti_pvt/./././././administrators.pwd*
Двойные слеши	*/_vti_pvt///administrators.pwd*
Аннулирование обхода	*/scripts/../_vti_pvt/administrators.pwd*

начальное положение. Эти методы могут сделать путь вторжения достаточно длинным, что не позволит СОВ обнаружить атаку.

Случай, когда можно декларировать переменные или осуществлять сравнения, более серьезен. Обычно это возможно для внутренних пользователей, которые уже имеют доступ к системе. В этом случае атакующий может модифицировать сигнатуру атаки, используя еле уловимые изменения. Например, можно спрятать зловредный код в переменную. Здесь кавычки разрушают сигнатуру, но, когда псевдоним выполняется, они удаляются и файл паролей высвечивается. Чтобы обнаруживать подобные атаки, СОВ должна интерпретировать команды аналогично командному интерпретатору (примеры приведены для Unix и Windows). Команды интерпретатора приведены в табл. 5.13.

Тот же подход может быть использован, если система допускает сравнения. Все, что нужно атакующему, это добавить сравнение, которое всегда истинно (например, AND 1 = 1), что модифицирует сигнатуру и может позволить обмануть СОВ. Этот метод часто используется в атаках SQL-вставок. Теоретически все приведенные методы сработают для приложений, базирующихся на текстах, таких как SNMP, SMTP, SQL запросы или telnet.

Таблица 5.13. **Команды интерпретатора**

Команда	Пример
Shell alias	*#Alias pass=rmore "/etc/"passwd'* *#pass*
Переменные окружения	*#test=/etc* *#more $test/passwd*
Переменные командной строки Windows	*C:\> set blah=c:\winnt\system32* *C:\> set extra=\cmd.exe* *C:\> %blah%%extra% /c dir c:*

Таблица 5.14. **Методы зловредного использования запроса**

Метод	Пример
Вставка нулевого знака	*GET%00/_vti_pvt/administrators.pwd*
Плохо форматированный запрос	*GET<tab>/_vti_pvt/administrators.pwd <tab>*
Спрятывание параметра	*GET /index.htm%3fparam=/../_vti_pvt/administrators.pwd*

Рассмотрим методы, которые работают только с запросами HTTP, так как предназначены для манипулирования с запрашиваемыми страницами. Рассмотрим нормальный HTTP-запрос:

```
GET /pages/index.htm HTTP/1.0
Header:…
```

В этом запросе четыре компонента, разделенных пробелами. Компонент *GET* — это метод, который «говорит» серверу, что мы передаем или получаем информацию. Компонент */pages/index.htm* — есть URL (Uniform Resource Identifier), который говорит Web-серверу, чего мы хотим. Компонент *HTTP/1.0* — номер версии HTTP, которую мы используем. Все после этого компонента является дополнительной информацией. Методы использования атакующим этих компонент приведены в табл. 5.14.

При вставке нулевого знака в запрос после указания метода хостовая СОВ будет видеть пустой GET-запрос, но некоторые серверы воспримут эту строку иначе. Другой метод состоит в замене пробела в каждой части запроса табуляцией. Большинство Web-серверов сжимают пробелы и возвращают значение, но сигнатурные хостовые СОВ могут не распознать разделитель (табуляцию) и не сгенерировать тревогу. Скрытие параметра в запросе основывается на том обстоятельстве, что хостовая СОВ может не проводить контроль параметров для повышения производительности (хостовые СОВ часто не исследуют то, что находится после знака вопроса, который обозначает, что далее идут параметры). Так как параметры будут различными для разных систем, сигнатурный метод может давать много ложных срабатываний. В приведенном примере главное находится после знака вопроса (%3f является ASCII кодом для ?, обозначающим начало списка параметров) и может не вызвать генерацию тревоги.

Перехват приложения (application hijacking) достаточно сложен, поэтому немногие хостовые СОВ могут его обнаружить. Сетевые СОВ обычно не рассматривают прикладной уровень, чтобы увидеть, что сеанс продолжен другим источником. Некоторые хостовые СОВ не просматривают стек, чтобы убедиться, что сеанс продолжен тем же источником.

Перехват приложений обычно формируется следующим образом. Пользователь начинает законный сеанс. Атакующий, определив это, делает невозможным продолжение работы машины законного пользователя и присваивает его атрибуты аутентификации. При использовании протокола без состояний (как HTTP) атакующему не нужно ограничивать законного пользователя, ему необходимо только иметь перехваченную метку сеанса. Это может быть достигнуто применением программ перехода на другой сайт (cross-site scripting) или атакой машины, содержащей метки сеанса (если машина пользователя недостаточно защищена).

Другим вариантом является метод «человек посередине». В этом случае атакующий претендует предстать законным пользователем для сервера и сервером для законного пользователя. После чего атакующий может модифицировать сеанс, чтобы не быть обнаруженным. Хостовые СОВ сконструированы для защиты ОС хоста, а не приложений или сетевых интерфейсов, поэтому могут обрабатывать приложения некорректно. Сетевые СОВ могут обнаружить подобные атаки, но большинство из них не исследуют все данные сеанса, например, чтобы контролировать правильность последовательных номеров.

5.7.3. Динамические методы обхода

Рассмотренные методы обхода СОВ являются простейшими (базовыми). Они известны с конца 90-х гг. XX в. и базируются в основном, на принципе ошибкоустойчивости (robustness). Этот принцип был закреплен еще в RFC 760, согласно которому реализация должна быть консервативна на передающей стороне и либеральна на приемной. Другими словами, программный код, посылающий команды или данные, должен строго соответствовать спецификации, а приемная сторона может принимать команды или данные даже не соответствующие спецификациям, если они понятны приемной стороне. Злоумышленники при проведении атак в этом случае выступают в роли отправителей, и, естественно, не выполняют правила. Современные прикладные протоколы сложны и позволяют различные интерпретации в применении. Поэтому злоумышленники используют редко применяемые свойства протокола в необычных комбинациях и приемная сторона либерально относится к интерпретации трафика, атака достигает назначения. Основные принципы современных методов обхода можно сформулировать следующим образом:

- формирование ввода «либеральным» путем (не соблюдая имеющиеся правила);
- применение консервативного метода приемными системами и устройствами защиты;
- использование редко применяемых свойств протоколов;

- использование необычных комбинаций параметров;
- использование существующих ограничений в устройствах безопасности (например, емкость памяти, ошибки проектирования и реализации и т. п.).

В настоящее время мы являемся свидетелями эры более изощренных хакерских методов, примерами которых стали успешные и необъясненные атаки против больших организаций. Анализ этих атак показал, что используются новые методы обхода. Компания Stonesoft (www.stonesoft.com) в 2010 г. открыла технологию новых методов обхода, которые были названы *динамическими методами обхода* (Advanced Evasion Technique, AET).

Суть этой технологии сводится к тому, что унаследованные или современные методы обхода при определенном использовании позволяют послать запрос на атакуемый узел таким образом, что средства сетевой защиты не смогут детектировать факт использования уязвимости или проявления аномальной активности. Ситуация усугубляется тем, что простейшие методы обхода можно комбинировать в самых разнообразных вариантах. На каждом из логических уровней модели TCP/IP существует свой набор протоколов, использующих формализованные структуры для передачи данных. Но каждое приложение использует цепочку вложенных протоколов различного уровня, на каждом из которых может быть применен метод обхода. Например, протокол ОС Windows — RPC (Remote Procedure Call, удаленный вызов процедур) реализуется цепочкой: NetBIOS — SMB — MSRPC. Но MSRPC может использовать не только транспорт NetBIOS для представления информации. Кроме того, вместо обычного SMB (1.0) в цепочке протоколов допустимо появление SMB2, равно как и передача может осуществляться поверх HTTP. Поэтому динамические методы обхода можно применять и комбинировать для каждого из используемых уровней. Например, к уже рассмотренным методам обхода на уровнях IP (фрагментация IP) и TCP (сегментация TCP) добавляется для SMB фрагментация и манипуляция транзакциями. Кроме того, могут использоваться такие методы, как случайные опции IP, флаг TCP urgent pointer и др. Применение динамических методов состоит в использовании особенностей представления данных на разных уровнях протоколов и приложений, в результате чего создается скрытый канал доставки содержимого пакета до целевого сервиса (приложения, системы), который не обнаруживается большинством современных СОВ.

Теоретическое количество возможных комбинаций использования методов обхода может составлять тысячи вариантов. В настоящее время компания Stonesoft уже обнаружила более 300 подобных методов.

Для того чтобы системы защиты смогли распознавать в сетевом потоке подобную нежелательную информацию, они должны определенным образом принимать и обрабатывать пакеты и содержащиеся

в них данные. Практически все современные средства защиты, работающие на уровне приложений, содержат в себе модули анализа трафика, которые вырабатывают сигналы тревоги по факту срабатывания некоторого правила. Большинство подобных систем использует анализ последовательности символов в потоке информации. Поэтому одной из важнейших задач обнаружения аномалий является корректный разбор и нормализация данных прикладных протоколов, т.е. модернизация механизмов инспекции трафика.

Компания Stonesoft предлагает решение для защиты от динамических методов обхода, заключающееся в разделении потока трафика на составляющие и проведении анализа на разных уровнях протоколов (нормализацией данных в зависимости от протокола, проводимой на различных уровнях).

5.8. Тестирование систем обнаружения вторжений

Вопросы тестирования СОВ еще недостаточно разработаны, не существует общепринятых методов и методик. Это вызвано тем, что для проведения тестирования СОВ помимо тестирования характеристик, которые связаны с принципами обнаружения, необходимо рассматривать и другие системные характеристики, например:

- время обнаружения (в реальном времени, не в реальном времени);
- градация поступления данных (непрерывно поступающие данные, данные, поступающие через регулярные интервалы времени);
- источники данных (сетевые данные, данные хоста — могут включать в себя записи аудита ОС, прикладных программ и системного оборудования);
- реакция на обнаруженное вторжение (пассивная — уведомление персонала, активная — изменение параметров защиты целевой системы или атака на нападающего);
- архитектура сенсоров системы (централизованная, распределенная);
- архитектура системы сбора данных (централизованная, распределенная);
- собственная безопасность (способность противостоять атаке против самой СОВ);
- степень совместимости с другими СОВ или другими системами обеспечения безопасности.

Кроме того, методы тестирования должны определяться назначением тестируемой системы. Поэтому можно говорить о двух различных видах тестирования: тестирование коммерческих (промышленных) систем и тестирование исследовательских прототипов.

5.8.1. Тестирование коммерческих систем

Различные организации, проводящие тестирование коммерческих систем (продуктов), используют свои собственные подходы. В качестве примера рассмотрим подход компании Denmac Systems, Inc.

Тестирование включает в себя рассмотрение количественных и качественных характеристик продукта.

К количественным характеристикам относятся:

- функциональное тестирование (цель — получение характеристик продукта, атакующий находится в другой сети);
- тестирование производительности (цель — определить характеристики производительности блока обработки пакетов СОВ, когда атакующий находится в той же сети).

Функциональное тестирование включает в себя восемь тестов.

1. Способность обнаружения множества атак — множество атак разбито на классы в соответствии с требуемым анализом различных частей стека TCP/IP. При тестировании рассматриваются атаки, которые связаны со свойством анализа заголовков (определяют способность СОВ обнаруживать атаки, связанные с заголовком IP-датаграммы, типичным примером такой атаки является атака Land), анализ свойств реассемблирования пакетов (определение возможности СОВ по реассемблированию множества фрагментированных пакетов и обнаружению атак, которые используют множество пакетов, примерами таких атак являются TearDrop и Ping of Death), атаки, связанные с анализом данных (анализируется способность СОВ обнаруживать атаку, содержащуюся в полезной нагрузке пакета, примером такой атаки является HTTP phf атака).

2. Защита СОВ от IP-десинхронизации — в этом случае атака маскируется путем применения нестандартных последовательностей и значений переменных.

3. Подавление тревог повторяющихся атак — оценивается способность сенсора генерировать тревоги и подавлять генерацию тревог при возникновении множества одинаковых атак за определенный промежуток времени.

4. Изменяемость фильтров — оценивается возможность приспособления текущих фильтров к нуждам пользователя и процедуры создания новых фильтров. При этом оцениваются возможность модификации или настройки существующих фильтров, возможность создания фильтров поиска заданной строки, возможность создания сложного фильтра, использующего возможности скриптов.

5. Генерация тревог — оценивается способность сенсоров генерировать и управлять сигналами тревоги, способность сигнализировать о возникновении тревоги; способность генерировать сигнал тревоги на внутреннее устройство (например, e-mail).

6. Регистрация — оцениваются свойства регистрации системы: способность сохранять результаты в различных форматах, способность конфигурировать записи регистрации для включения в них различных переменных.

7. Генерация отчетов — оценивается способность генерации отчетов для различных тревог: генерация отчетов об активности, способность приспособления к запросам, способность объединения отчетов по заданным переменным, способность создания и сохранения отчетов, выполненных по заказу (профилей).

8. Распределенность архитектуры — оценивается способность формирования распределенной архитектуры, необходимой для больших корпоративных сетей. Оцениваются: архитектура (одиночная или распределенная), способность множества сенсоров передавать отчеты центральной консоли управления, способность применения иерархической модели для сенсоров и консолей управления.

Тестирование производительности проводилось в сети с пропускной способностью 100 Мбит/с при следующих характеристиках трафика:

- тесты формировались со скоростью 6 100 пакетов/с (3 % пропускной способности канала);
- тесты формировались со скоростью 25 000 пакетов/с (15 % пропускной способности канала).

Все тесты использовали 64-байтовые пакеты.

Атаки формировались специальными средствами и посылались с различной скоростью: со скоростью формирования пакетов, со скоростью 100 пакетов/с.

Для оценки производительности используются следующие тесты.

1. Скорость обработки — оценивается способность анализа пакетов без пропусков пакетов. Оценка проводится при отсутствии атак (сенсоры не содержат сигнатур атак).

2. Реассемблирование пакетов — оценивается производительность реассемблирования пакетов (используется атака Ping of Death, сенсоры содержат ее сигнатуру).

3. Эффективность фильтрации — оценивается эффективность фильтров по обнаружению атак и генерации сигналов тревоги (используется атака Land).

Качественные характеристики (оценка удобства) включают в себя следующие шесть критериев.

1. Удобство интерфейса — оценивается удобство, полнота и возможность расширения интерфейса пользователя. Учитываются возможность поддержки различных ОС, легкость использования и стабильность работы.

2. Реализация в виде устройства или программы — оценивается степень интеграции в ОС с точки зрения простоты обслуживания и эксплуатации. Более высоко оцениваются продукты со встроенной ОС, чем те, которые опираются на существующие ОС.

Таблица 5.15. **Протестированные продукты**

Обозначение	Продукт	Компания
NFR	Network Flight Recorder v.4.0	NFR Security, Inc.
RS	RealSecure v.3.0	Internet Security Systems, Inc.
SW	SessionWall v.3.1	Computer Associates International, Inc.
NP	NetProwler v.3.0	Axent Communications, Inc.
NR	NetRanger v.2.2	Cisco Systems, Inc.

Таблица 5.16. **Результаты тестирования**

№ теста	Критерий	Вес	NFR	RS	SW	NP	NR
	Функциональное тестирование						
1	Обнаружение множества атак	5	4	4	2	1	4
2	Защита СОВ от IP-десин-хронизации	3	4	1	0	0	N/A
3	Подавление тревог повторяющихся атак	3	1	0	4	0	3
4	Изменяемость фильтров	5	4	1	2	3	2
5	Генерация тревог	4	3	3	4	3	3
6	Регистрация	4	4	3	3	3	4
7	Генерация отчетов	3	2	3	4	3	3
8	Распределенность архитектуры	3	2	3	4	3	3
	Тестирование производительности						
1	Скорость обработки	3	3	3	1	4	N/A
2	Реассемблирование пакетов	5	2	4	N/A	0	N/A

Окончание табл. 5.16

№ теста	Критерий	Вес	NFR	RS	SW	NP	NR
3	Эффективность фильтрации	5	2	3	N/A	2	N/A
Взвешенные оценки по двум классам критериев			2,00	2,64	2,41	1,79	2,90
Оценка удобства							
1	Удобство интерфейса	3	3	3	4	2	2
2	Устройство/программа	4	4	1	1	1	3
3	Зрелость продукта	2	3	4	4	1	3
4	Концепция продукта	1	3	3	4	4	4
5	Опыт компании	3	4	4	2	3	2
6	Цена продукта	4	4	3	2	3	1
Взвешенная оценка продукта			3,03	2,69	2,43	1,90	2,84

3. Зрелость продукта — оцениваются следующие параметры: стабильность, реализация, полнота и точность выполнения объявленных функций, время существования на рынке.

4. Концепция продукта — оцениваются достоинства и уникальные свойства продукта.

5. Опыт компании — оценивается назначение компании и ее вклад в развитие технологий обеспечения безопасности, особенно в области обнаружения вторжений. Оценивается участие компании в исследованиях и разработке новых безопасных технологий.

6. Цена продукта — определяются и ранжируются стоимость продукта и его эксплуатации.

Для сравнения различных СОВ в результате рассмотрения каждый критерий получает оценку из диапазона от 0 до 4 (4 — высшая оценка).

Для того чтобы сравнивать критерии, относящиеся к различным классам оценивания, вводятся веса соответствия (или важности) оцениваемого показателя. Веса назначаются в интервале от 1 до 5 (5 — наиболее важный критерий).

В результате подсчитывается объединенная оценка продукта по следующей формуле:

$$\text{Взвешенная оценка продукта} = \frac{x_i w_i}{\sum w_i},$$

где x_i — оценка критерия; w_i — вес важности критерия.

Оценке по данной методике в 2000 г. подверглись продукты, перечисленные в табл. 5.15.

Значения весов для приведенных критериев и оценки для этих продуктов приведены в табл. 5.16.

Отсутствие значений в таблице (параметр N/A) показывает невозможность оценки данного критерия или невозможность применения данного теста.

5.8.2. Тестирование исследовательских прототипов

В 1998 г. по заданию DARPA (Defence Advanced Research Project Agency) лаборатория Линкольна Массачусетского технологического института (MIT Lincoln Laboratory) провела оценку различных СОВ. Во время этой оценки исследователи использовали данные сенсоров в виде записанного сетевого трафика, записи аудита Solaris BSM (записи Windows NT были добавлены в 1999 г.), а также данные состояния файловых систем, чтобы идентифицировать вторжения, которые были проведены для тестовой сети во время сбора данных. Тестовый трафик (тренировочные данные) состоял из смеси реального и моделируемого фонового трафиков, а атаки были направлены на реальные машины тестовой сети. Тренировочные данные содержали множество атак, которые были идентифицированы в сопутствующих документах.

Анализ полученных оценок в 1998 г. показал множество серьезных недостатков в оцениваемых СОВ. Поэтому оценивание было повторено в 1999 г. После этого было принято решение о сохранении тренировочных данных и свободном доступе к ним со стороны исследователей. В настоящее время эти тренировочные данные доступны для всех желающих. Это дает возможность исследователям практически оценить важнейшие характеристики, связанные с обнаружением, а именно: вероятности ошибок ложного срабатывания (false positive) и ошибок пропуска (false negative).

Общее число типов атак, включенных в тестовые данные 1998 г., составило 32. Эти атаки, с точки зрения атакующего, можно подразделить на четыре категории:

- атаки отказа в обслуживании (тип DoS, Denial of Service);
- атаки перехода от удаленного использования к локальному (тип R2L, Remote to Local);
- атаки получения пользователями прав суперпользователя (тип U2R, User to Root);
- атаки сканирования или проб (тип PRB, Probing/surveillance).

В тренировочных записях, которые были дополнены и скорректированы в 1999 г., содержатся реальные записи аудита и записи сетевого трафика в формате tcpdump, которые представляют собой записи шести недель тренировочных данных и двух недель тестовых данных.

Положительный опыт предоставления исходных данных для исследователей и разработчиков систем обнаружения вторжений был продолжен. В настоящее время существует несколько подобных общедоступных баз данных тестовых записей.

5.8.3. Методы формирования тестовых наборов

Для проверки (оценки) СОВ необходимо подготовить два основных множества исходных данных: данные для обучения (тренировки) СОВ и данные для проверки (тестирования). Рассмотрим основные методы формирования таких множеств.

Метод удержания (holdout) используется в тех случаях, когда объем данных, которые можно использовать для тренировки и тестирования, достаточно велик. Тогда весь набор данных случайным образом разделяется на два непересекающихся множества: тренировочное и тестовое. Обучение СОВ (классификатора) проводится на тренировочном множестве, а проверка — на тестовом. Обычно для тестирования отводится от 1/10 до 1/3 количества всех доступных записей набора данных. Для получения более надежной оценки подобная процедура (удержания части данных) проводится несколько раз и полученные результаты усредняются (repeated holdout).

Метод стратификации (stratification) применяется в том случае, когда в доступных данных объекты одного класса встречаются значительно чаще объектов другого класса (один класс доминирует над другим, например, нормальное поведение значительно превышает количество атак). В этом случае в тренировочное (или тестовое) множество может попасть малое количество экземпляров одного из классов или даже вообще ни одного экземпляра. Процедура стратификации заключается в том, что при разделении множества данных на тестовое и тренировочное данные каждого класса разделяют в нужной пропорции и затем из полученных четырех множеств формируют тестовое и тренировочное. Это позволяет получить достаточную представительность каждого класса, как на этапе тренировки, так и при тестировании.

Метод перекрестной проверки (cross-validation) заключается в разделении всего набора данных на k подмножеств (можно с использованием метода стратификации). Затем проводится обучение (тренировка) по $k - 1$ подмножествам с тестированием по одному оставшемуся подмножеству. Результаты, полученные на каждой из k итераций, усредняются для получения конечного результата. Обычно k принимают равным 10 (tenfold cross-validation).

5.8.4. Матрица несоответствий

Для проведения оценки СОВ по подготовленным тестовым наборам (множествам) составляется так называемая матрица несоответствий (confusion matrix). Пример матрицы несоответствий для случая разделения на два класса (норма и атака) приведен на рис. 5.17.

Принято считать, что основная гипотеза H_0 соответствует состоянию «по умолчанию», представляющему естественный, наиболее ожидаемый порядок вещей. В нашем случае это нормальный трафик (отсутствие атак). Противоположной гипотезой H_1 будет атака (трактуемая как менее вероятное событие, требующее соответствующей реакции). Тогда случай, когда система обнаружения определяет наличие атаки (H_1) при ее отсутствии (H_0), называют ошибкой 1-го рода или ложной тревогой, ложным срабатыванием. Симметричный случай (когда при наличии атаки система не обнаруживает ее) называют ошибкой 2-го рода или пропуском события.

Построение такой матрицы несоответствий позволяет определить ряд вспомогательных и основных параметров оценки качества обнаружения:

- доля верных положительных классификаций (true positive rate, *TPR*), $TPR = TN/(FP + TN)$;
- доля ложных положительных классификаций (false positive rate, *FPR*), $FPR = FN/(TP + FN)$;
- доля отрицательных случаев, которые правильно определены (true negative rate, *TNR*), $TNR = TP/(TP + FN)$;
- доля положительных случаев, которые некорректно отнесены к правильным (false negative rate, *FNR*), $FNR = FP/(FP + TN)$.

Данные параметры позволяют судить о полученных долях ошибок 1-го и 2-го рода при проведении тестирования.

Основными параметрами, применяемыми для оценки СОВ, являются чувствительность (precision) и точность (accuracy).

Чувствительность — это отношение числа правильно определенных атак к сумме всех определений атак:

$$P = TN/(FN + TN).$$

В действительности	Результат обнаружения	
	Норма	Атака
Норма	*TP*-норма определена верно	*FN*-норма определена неверно
Атака	*FP*-атака определена неверно	*TN*-атака определена верно

Рис. 5.17. Матрица несоответствий

Точность определяют по следующей формуле:

$$A = (TP + TN) / (TP + TN + FP + FN).$$

Данные параметры дают возможность сравнить различные СОВ (или их классификаторы).

Алгоритм сравнения классификаторов, используемых в СОВ, можно представить несколькими шагами:

1) выбрать метод разделения множества данных на тренировочное и тестовое множества;

2) провести испытания различных классификаторов на одинаковых наборах;

3) определить численные характеристики;

4) выбрать классификатор с лучшими характеристиками.

Простейшим способом сравнения двух классификаторов является сравнение их уровней ошибок 1-го и 2-го рода.

Выбор оптимального классификатора проводится, исходя из требований к поставленной задаче. Например, определить, что важнее — уменьшить уровень ошибок первого или второго рода.

В случае когда классификатор содержит свободный параметр, позволяющий регулировать соотношение *TPR* и *TFR*, целесообразно использовать так называемую оперативную характеристику.

Оперативная характеристика (ГОСТ Р 50779.10—2000) — функция, которая определяет вероятность принятия гипотезы H_0 относительно значений скалярного параметра.

Кривая оперативной характеристики (receeving operating curve, ROC) представляет собой график, в котором по оси абсцисс откладывается *FPR*, а по оси ординат — *TPR*. Пример кривой оперативной характеристики приведен на рис. 5.18. На этом рисунке пунктиром показана прямая, соответствующая равновероятному выбору классификатора. Точка (0, 1) соответствует наилучшему классификатору — он определяет все положительные и отрицательные случаи корректно. Точка (0, 0) представляет собой классификатор, который говорит, что все классы отрицательны. Точка (1, 1) обозначает классификатор, который считает, что каждый класс положителен. Точка (1, 0) — классификатор с неправильной классификацией. Во многих случаях классификатор имеет параметр, который можно изменять для увеличения *TP* как стоимости увеличения *FP*. Или для уменьшения *FP* как стоимости уменьшения *TP*. Каждое значение параметра дает пару (*FP*, *TP*), а серия таких пар — ROC-кривую. Непараметрический классификатор представляется единственной точкой, соответствующей паре (*FP*, *TP*).

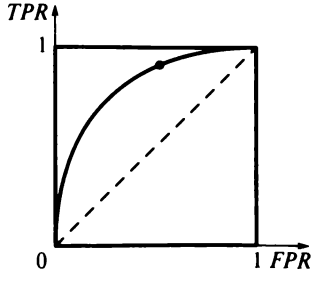


Рис. 5.18. Вид кривой оперативной характеристики

5.9. Системы предупреждения вторжений

Принятая в настоящее время концепция эшелонированной обороны корпоративных сетей в качестве второго уровня предусматривает обнаружение наличия атак в трафике, разрешенном межсетевым экраном (первый уровень), и защиту от них. Эти функции возлагаются на системы обнаружения вторжений, которые в силу ряда причин не могут обеспечить достаточно высокий уровень защиты.

Современные СОВ характеризуются достаточно большим числом ложных срабатываний (false positives), что требует значительных затрат времени и ресурсов для анализа причин возникновения тревог, их достаточно трудно инсталлировать, обновлять и эксплуатировать. Управление СОВ требует высокой квалификации администраторов системы.

Возрастает число атак, направленных на обход или обман СОВ. Наиболее часто для этого используются подмена путей, кодирование знаков и искусственная фрагментация. Кроме того, появились полиморфные атаки, которые могут обмануть большинство СОВ.

За последние годы число и масштабы корпораций значительно выросли. Если несколько лет назад система с 50 сенсорами считалась большой, то в настоящее время многие корпорации требуют установки сотен и даже тысяч сенсоров. Поэтому современные СОВ должны учитывать постоянную расширяемость сетей корпораций. Считается, что использование полосы пропускания растет экспоненциально, большие корпорации уже сейчас используют гигабитные сети. Поэтому СОВ должны обеспечивать функционирование при пиковых нагрузках габитных сетей без потери какой-либо части трафика.

То, что СОВ является пассивным устройством, является их ахиллесовой пятой. Поэтому необходим переход к активным функциям. В настоящее время обнаружение атак ценится ниже их предупреждения.

Пассивные средства всегда являются уязвимыми к различным атакам обхода (обмана). Поэтому исследователи и компании-производители начали разработку и производство активных средств, которые получили название средств предупреждения вторжений (СПВ). Первоначально применяли простейшие механизмы, которые могли использовать СОВ для остановки обнаруженной атаки: посылка пакета TCP Reset или выдача активному устройству (маршрутизатору или межсетевому экрану) сигнала на блокировку соответствующего трафика, но сложности применения этих методов привели к необходимости создания нового типа активных устройств защиты — активных систем предупреждения вторжений.

Посылка пакета TCP Reset. При обнаружении атаки устройство посылает пакет TCP Reset в оба направления, что разрывает

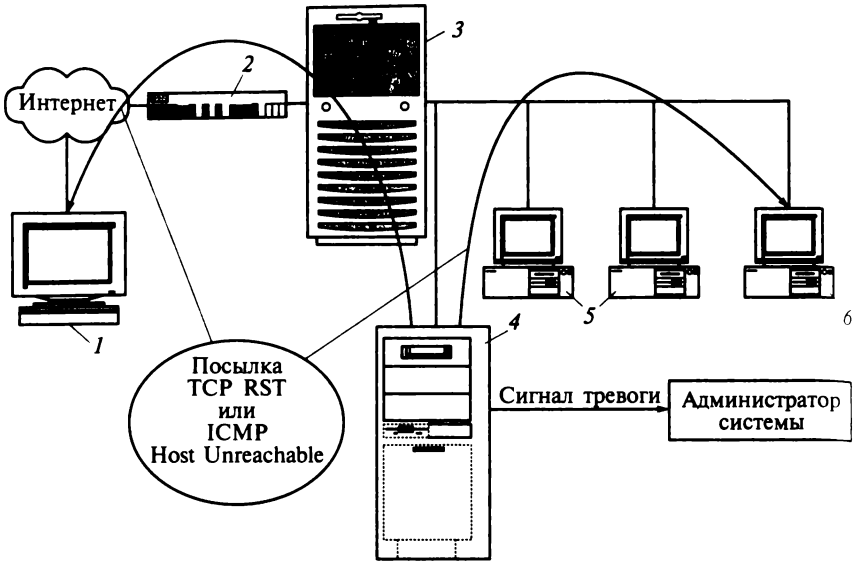


Рис. 5.19. Посылка пакета TCP Reset:

1 — атакующий; *2* — маршрутизатор; *3* — межсетевой экран; *4* — система предупреждения вторжений; *5* — хост; *6* — атакуемый хост

соединение и препятствует развитию атаки (рис. 5.19). Данный подход, несмотря на свою кажущуюся простоту, обладает следующими недостатками:

- этот подход применим только к протоколу TCP или прикладным протоколам, базирующимся на TCP. Одним из вариантов преодоления данного недостатка является посылка сообщения ICMP Host unreachable, что, в свою очередь, вызывает проблемы;
- при посылке TCP Reset необходимо указать соответствующий последовательный номер пакета. Если это легко сделать для пакета, направленного на защищаемый хост, то для посылки пакета атакующему хосту необходимо найти соответствующий последовательный номер (или сохранять эти номера в устройстве обнаружения);
- существует временная задержка, которая складывается из времени обработки пришедшего пакета, принятия решения о наличии атаки, формирования пакета TCP Reset и времени передачи пакета к месту назначения. За это время атака может быть реализована.

Выдача сигнала фильтрующему устройству. В этом случае устройство обнаружения атаки сигнализирует маршрутизатору или МЭ о наличии атаки, чтобы была заблокирована соответствующая часть трафика путем введения дополнительных правил фильтрации (рис. 5.20). Реализация данного подхода также связана с определенными трудностями:

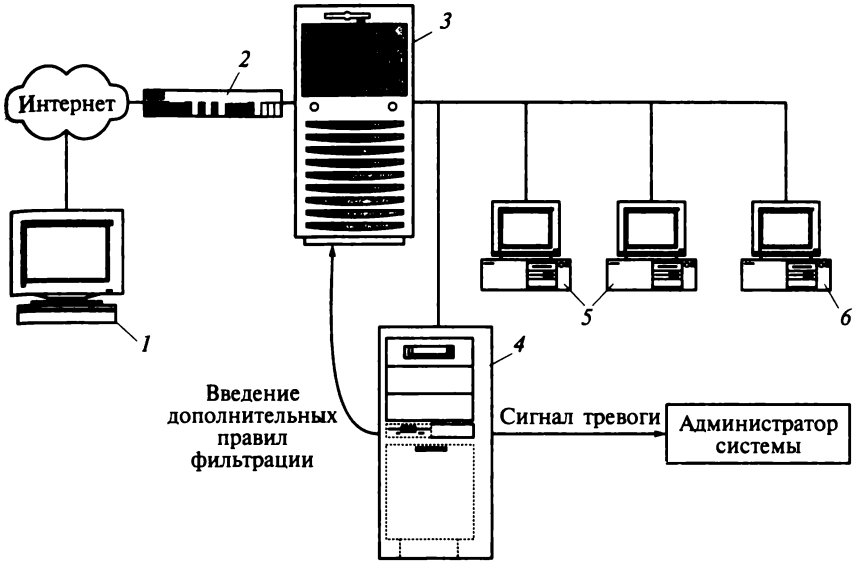


Рис. 5.20. Выдача сигнала фильтрующему устройству:

1 — атакующий; *2* — маршрутизатор; *3* — межсетевой экран; *4* — система предупреждения вторжений; *5* — хост; *6* — атакуемый хост

• межсетевой экран должен иметь возможность принимать соответствующие сигналы и оперативно изменять свои фильтрующие правила в зависимости от полученного сигнала (таким свойством обладает, например, Check Point FireWall-1);

• межсетевой экран должен «знать», какую часть трафика необходимо заблокировать. Это реализуется путем блокировки соответствующего IP-адреса, что (в случае поддельного адреса источника) может привести к отказу в обслуживании;

• временной промежуток от получения пакета атаки до блокирования соответствующего трафика, которого может быть достаточно для успеха атаки.

Активные системы предупреждения вторжений. Эти системы могут удалять пакет, если считают его источником атаки (рис. 5.21).

Кроме того, они должны обеспечить минимальное число ложных тревог, так как будут блокировать трафик, признанный зловредным. Поэтому такие системы должны базироваться на технологиях, обеспечивающих высокую точность обнаружения атак. Одним из важнейших показателей для таких систем является производительность. Системы предупреждения вторжений должны обеспечить функционирование с такой скоростью, чтобы не снизить имеющуюся пропускную способность канала. Это приводит к тому, что современные СОВ строятся с использованием специализированных для приложе-

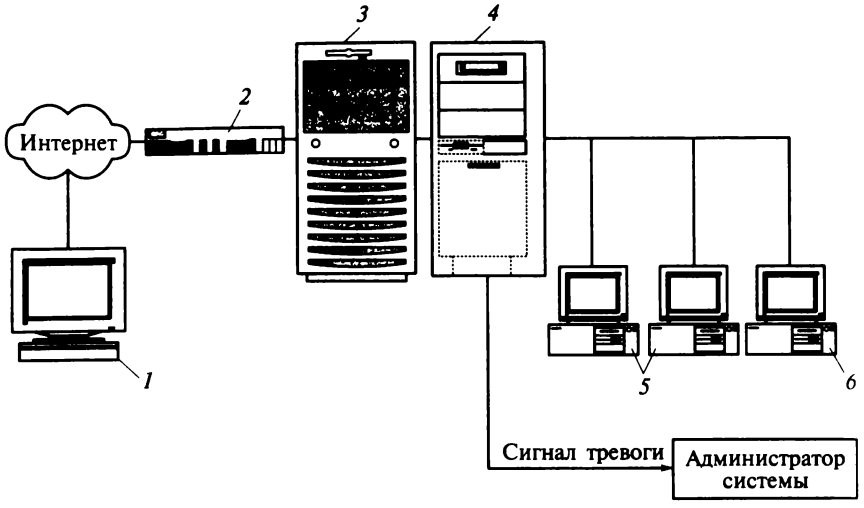


Рис. 5.21. Активная система предупреждения вторжений:

1 — атакующий; *2* — маршрутизатор; *3* — межсетевой экран; *4* — система предупреждения вторжений; *5* — хост; *6* — атакуемый хост

ний интегральных схем (Application Specific Integrated Circuits, ASIC) и программируемой логики (Field-Programmable Gate Array, FPGA).

Термин «предупреждение вторжений» можно рассматривать как широкое понятие, охватывающее множество свойств, которые уже существуют в устройствах защиты, таких как антивирусная защита, межсетевые экраны и системы обнаружения вторжений. Основные продукты обнаружения вторжений, как правило, используют технологию обнаружения сигнатур. В качестве примера рассмотрим подходы, использованные в системе Attack Mitigator IPS компании Top Layer Networks, Inc. (http://www.TopLayer.com/).

Данная система использует ASIC устройства для обеспечения высокой скорости обработки. Собственно система представляет собой многоуровневую архитектуру, на каждом уровне которой принимается решение об отнесении анализируемой части трафика к одному из трех состояний: «хорошее» — трафик передается на последний уровень; «зловредное» — трафик удаляется и «подозрительное» — трафик передается для обработки на следующий уровень. Архитектура включает в себя следующие пять уровней:

1) анализ трафика на уровне каждого сеанса — для каждого сеанса создаются записи в таблицах сеансов. Контроль сеансов позволяет обнаруживать атаки, реализуемые одним пакетом, и атаки, использующие фрагментацию;

2) осуществление блокировки атак отказа в обслуживании и распределенных атак отказа в обслуживании (Distributed Denial of Service.

DDoS). Для этого используется оперативная память для 200 000 IP-адресов текущих соединений (чтобы определить, например, завершение установления TCP соединения);

3) защита от червей и Web-приложений, функционирующих на 80-м порту. Для защиты от червей используются специальные алгоритмы анализа состояний, идентифицирующие варианты существующих эксплойтов. Для защиты протокола HTTP используется механизм нормализации протокола, который, в частности, декодирует URI (Uniform Resource Identifier), чтобы обнаружить возможные атаки;

4) обнаружение аномалий протоколов и аномалий трафика на основе заранее установленных порогов. Пороги устанавливаются администратором системы или автоматически в процессе нормальной работы системы. Кроме того, на данном уровне можно определить число возможных полуоткрытых сеансов, таких, например, как TCP-соединений;

5) обработка всего подозрительного трафика, не пропущенного предыдущими уровнями. На этом уровне определяется стратегия реакции системы в зависимости от установленной политики безопасности организации.

Данный подход в той или иной степени поддерживается и другими производителями систем предупреждения вторжений. Дополнительным свойством таких систем может быть использование корреляции между событиями, обнаруживаемыми в различных уровнях.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите основные причины невозможности обеспечения адекватной защиты только средствами межсетевых экранов.
2. Перечислите основные проблемы формирования сигнатур атак.
3. Дайте определение понятий «атака» и «вторжение».
4. Что характеризуют профили в модели Д. Деннинг?
5. Каково основное отличие между профилями и шаблонами профилей в модели Д. Деннинг?
6. Что понимается под аномальностью в модели Д. Деннинг?
7. Объясните наличие в модели CIDF двусторонних связей.
8. Что включается в понятие «сигнатуры СОВ»? Какие виды сигнатур могут использоваться?
9. Перечислите задачи, выполняемые системами обнаружения аномалий протоколов.
10. Какие подходы могут использоваться для определения «нормальности» поведения?
11. Дайте определение понятия data mining.
12. Укажите основные достоинства применения технологии агентов для обнаружения вторжений.
13. Почему при реализации генетических алгоритмов применяются малые значения вероятности мутации?

14. Какие подходы используются для нахождения генетическими алгоритмами локальных максимумов?
15. Предположим, что СОВ реализует модель CIDF с использованием аппарата нейронных сетей. Какие связи в модели CIDF нуждаются в изменении?
16. Чем определяется выбор размерности для входного слоя нейронной сети?
17. Перечислите основные методы обхода систем обнаружения вторжений.
18. В чем состоит основная цель Honeypot и Honeynet?
19. Что такое десинхронизация СОВ?
20. Укажите причины появления ошибок ложного срабатывания и ошибок пропуска для различных технологий обнаружения: обнаружения сигнатур и обнаружения аномалий.
21. Укажите принципиальное отличие систем предупреждения вторжений от систем обнаружения вторжений.

Глава 6

ВИРТУАЛЬНЫЕ ЧАСТНЫЕ СЕТИ

— С позволения Вашего Величества, — сказал Валет, — я этого письма не писал, и они этого не докажут. Там нет подписи.
— Тем хуже, — сказал Король. — Значит, ты что-то дурное задумал, а не то подписался бы, как все честные люди.

Л. Кэрролл. Алиса в Зазеркалье

Виртуальные частные сети, или защищенные виртуальные сети (Virtual Private Network, VPN), — это подключение, установленное по существующей общедоступной инфраструктуре и использующее шифрование и аутентификацию для обеспечения безопасности содержания передаваемых пакетов.

Виртуальная частная сеть создает виртуальный сегмент между любыми двумя точками доступа к сети. Она может проходить через общедоступную инфраструктуру локальной вычислительной сети, подключения к глобальной сети (Wide area Network, WAN) или Интернет.

Рассмотрим VPN, организующие безопасные каналы передачи данных, использующие общедоступную инфраструктуру или Интернет. Все VPN по конфигурации можно подразделить на три основных типа:

1) узел-узел (host-to-host);
2) узел-шлюз (host-to-gateway);
3) шлюз-шлюз (gateway-to-gateway).

Для организации канала связи, проходящего через Интернет, можно использовать VPN любого типа.

Основной концепцией VPN является защита шифрованием канала связи на различных уровнях модели TCP/IP, а именно:

- прикладном (5-й уровень);
- транспортном (4-й уровень);
- сетевом (3-й уровень);
- канальном (2-й уровень).

Схема расположения протоколов VPN по уровням модели приведена на рис. 6.1.

На *прикладном уровне* шифрование можно применять с помощью программ, подобных пакету Pretty Good Privacy (PGP), или через каналы типа Secure Shell (SSH). Такие программы работают на участке сети от узла до узла, что означает, что они предлагают защиту только для содержимого (payload) пакета, а не всего пакета в целом.

Уровень TCP/IP	Основные протоколы
Прикладной (application)	PGP, S/MIME SSH, Kerberos, RADIUS
Транспортный (transport)	SSL, TSL SOCKS v5
Сетевой (internetworking)	IPSec (AH, ESP)
Канальный (datalink)	L2TP, PPTP, L2A CHAP, PAP, MS-CHAP

Рис. 6.1. Схема расположения протоколов VPN по уровням модели

Исключение составляет протокол SSH, который может использовать режим port-forwarding для создания туннеля.

На *транспортном уровне* для защиты содержимого пакетов конкретного сеанса между двумя сторонами можно использовать протоколы, аналогичные протоколу защищенных сокетов (Secure Sockets Layer, SSL). Обычно такой метод используется при соединениях, установленных посредством Web-браузера. При этом также защищается только содержательная часть передаваемых пакетов, а IP-датаграммы, которые несут эту информацию, доступны для просмотра.

На *сетевом уровне* протоколы, подобные IPSec, не только зашифровывают содержательную часть пакета (полезную нагрузку), но и зашифровывают информацию заголовков протоколов TCP/IP.

На *канальном уровне* протокол туннелирования (Layer 2 Tunneling Protocol, L2TP) является расширением протокола соединения типа «точка-точка» (Point-to-Point Protocol, PPP), который допускает шифрование пакетов, посланных по PPP-протоколу на канальном уровне передачи данных.

Несмотря на то что эти технологии шифрования применяются на разных уровнях, они все могут быть частью VPN. Необходимо отметить, что некоторые из этих технологий не могут обрабатывать все режимы работы VPN без дополнительной помощи со стороны других приложений или протоколов.

6.1. Туннелирование

Туннелирование — это процесс инкапсуляции одного типа пакетов внутри другого в целях получения некоторого преимущества при транспортировке. Сама идеология построения стека TCP/IP показы-

вает пример туннелирования. Например, туннелирование можно использовать, чтобы послать трафик через маршрутизируемую сетевую среду или чтобы применить шифрование для обеспечения безопасности IP-пакетов. В качестве примера рассмотрим VPN типа шлюз-шлюз (рис. 6.2).

В данном примере МЭ преобразует все пакеты, предназначенные для удаленной сети, в зашифрованный вид и добавляет к ним новые IP-заголовки со своим собственным IP-адресом в качестве отправителя и адресом удаленного МЭ в качестве IP-адреса назначения.

В этом случае шифрование скрывает фактическую информацию, содержащуюся в оригинальном IP-пакете. Когда удаленный МЭ получает пакет, он расшифровывает его и передает узлу сети, для которого он предназначался. Виртуальный сегмент сети, создаваемый между двумя шлюзовыми оконечными точками, называется туннелем (так как конечные узлы удаленных локальных сетей «не имеют представления» о том, что происходит с их пакетами во время доставки).

Ниже приведена запись в формате tcpdump пакета, не проходящего через зашифрованный туннель.:

```
00:05:18.671517 192.168.44.129 > 172.16.1.128: AH
%spi=580532459, seq=0x3): 1232 > 80: P 1:260 (259) ack 1
%win 17520 (DF)
```

Фактически эта запись представляет собой пакет IPSec-протокола, использующий протокол аутентификации заголовка (Authentification Header, AH). Этот пакет посылается в режиме, не требующем туннелирования. Пакет проходит от одного узла сети к другому, не будучи

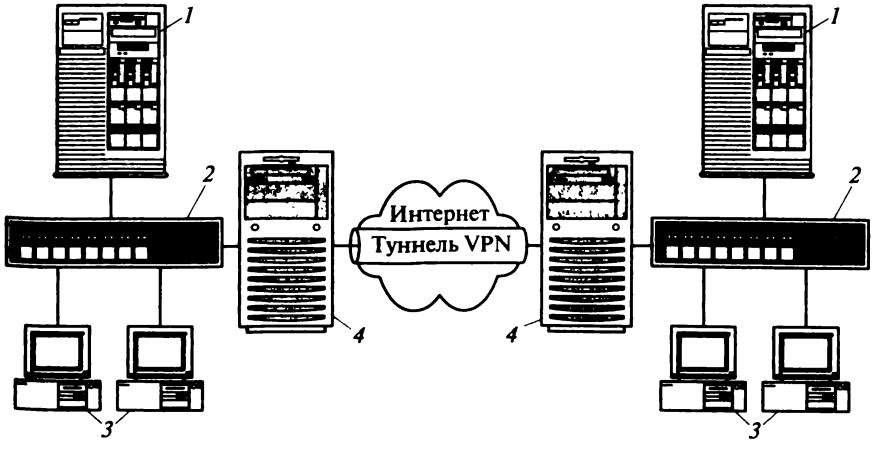


Рис. 6.2. Пример построения VPN типа шлюз — шлюз:
1 — сервер; *2* — концентратор; *3* — рабочая станция; *4* — межсетевой экран

транслированным шлюзовыми устройствами. В данном примере IP-адреса соответствуют адресам конкретных узлов, кроме того, доступна информация транспортного уровня (пакет является транзакцией HTTP-протокола); значения флагов показывают, что это фактически окончание установления соединения с квитированием. Ниже приведен выход tcpdump, соответствующий инкапсулированному пакету (та же транзакция, но транслированная туннелем VPN типа шлюз-шлюз):

```
00:01:30.057912 192.168.44.1 > 172.16.1.1
ESP(spi=1302500357, seq=0x3) (DF)
```

В данном случае видны только IP-адреса шлюзов и то, что применен протокол безопасной инкапсуляции содержимого пакета (Encapsulating Security Payload, ESP).

Главная выгода от использования VPN для удаленного доступа — совокупность стоимостной эффективности возможного использования общедоступной сетевой среды для транспортирования частной информации и высокого уровня безопасности. Защищенная виртуальная сеть может предоставить множество уровней безопасности, включая усовершенствование конфиденциальности, целостности и аутентификации. Поскольку VPN использует существующую сетевую инфраструктуру, ее можно реализовать достаточно быстро, так как нет необходимости прокладывать новые (выделенные) линии связи.

Комбинация безопасности, быстрой установки и рентабельности с точки зрения стоимости может сделать VPN превосходным коммерческим коммуникационным решением.

Несмотря на все свои положительные стороны, VPN не избавлены от недостатков. Среди них необходимо отметить следующие:

- накладные расходы обработки данных — в основе VPN лежит использование шифрования, требующее выполнения большого числа операций. Это сказывается не только на шлюзовых устройствах, но и на общей пропускной способности подключения VPN;
- пакетные накладные расходы, вызываемые добавлением заголовков к каждому пакету, что, в свою очередь, может вызвать необходимость фрагментации пакетов;
- проблемы реализации, связанные с применением трансляции сетевых адресов (NAT) для VPN, с размером максимального блока передачи данных (Maximum Transmission Unit, MTU) и т. д.;
- проблемы управления и поиска конфликтов, поскольку внутренние части структуры и содержимое инкапсулированных пакетов недоступны до момента их расшифровки;
- проблемы с функционированием сетевых систем обнаружения вторжений (закрытие содержимого пакетов);
- проблемы Интернета, повреждения канала связи с провайдером или атаки отказа в обслуживании, направленные на шлюз VPN.

6.2. Протоколы VPN канального уровня

На канальном уровне существуют два протокола для реализации VPN: протокол туннелирования типа точка—точка (Point-to-point Tunneling Protocol, PPTP) и протокол туннелирования второго уровня (Layer Two Tunneling Protocol, L2TP). Оба этих протокола включены в состав операционной системы Microsoft Windows.

Протокол PPTP. Протокол PPTP является дальнейшим развитием протокола PPP, который распространился в связи с появлением модемного доступа к сети Интернет. Протокол PPTP был разработан консорциумом таких производителей, как Microsoft, US Robotics, Ascend и 3com. Для шифрования протокола PPP был использован протокол двухточечного шифрования Microsoft (Microsoft Point-to-Point Encryption, MPPE), который использует алгоритм RC4. Однако большинство проблем в области безопасности было связано с ненадежностью используемого метода аутентификации — протокола Microsoft аутентификации с предварительным согласованием вызова (Microsoft Challenge/Reply HandShake Protocol, MSCHAP). Для устранения недостатков был выпущен протокол MSCHAP версии 2. Протокол PPTP использует все связанные протоколы, которые подобны протоколу MSCHAP, протоколу аутентификации пароля (Password Authentication Protocol, PAP), протоколу аутентификации с предварительным согласованием вызова (Challenge Handshake Authentication Protocol, CHAP), расширенному протоколу аутентификации (Extensible Authentication Protocol, EAP).

Протокол PPTP использует два канала, работающих совместно. Первый — канал управления (порт 1723/tcp). Этот канал посылает в обе стороны все команды, которые управляют сеансом подключения. Второй — инкапсулированный канал передачи данных, являющийся вариантом протокола общей инкапсуляции для маршрутизации (Generic Routing Encapsulation, GRE). Это протокол 47, который использует UDP в качестве транспортного протокола.

Преимуществом туннеля протокола общей инкапсуляции для маршрутизации является то, что он может инкапсулировать и передавать протоколы, отличающиеся от протокола IP.

Протокол PPTP работает без помех через устройства NAT. Он интегрируется со многими аппаратными устройствами. Протокол PPTP при инициализации связи использует протокол PPP, поэтому может оказаться уязвимым к атакам типа spoofing и «человек посередине».

Протокол L2TP. Протокол L2TP определен в документе RFC 2661 и фактически является гибридом двух предыдущих протоколов туннелирования: протокола пересылки второго уровня (Layer Two Forwarding, L2F) компании Cisco и протокола PPTP. Он заменил протокол PPTP в качестве решения для VPN в операционной системе Windows 2000.

Протокол L2TP, подобно протоколу PPTP, использует при аутентификации пользователя возможности протокола PPP (протоколы MSCHAP, CHAP, EAP, PAP и т.д.). Аналогично протоколу PPTP протокол L2TP использует два канала связи: сообщения управления и сообщения туннеля для передачи данных. Первый бит заголовка протокола PPTP служит для опознания этих типов сообщений (1 — для сообщений управления, 0 — для сообщений данных). Сообщениям управления дается более высокий приоритет по отношению к сообщениям данных, чтобы гарантировать, что важная информация администрирования сеанса будет передана настолько быстро, насколько это возможно.

Подключение канала управления устанавливается для туннеля, который затем сопровождается инициированием сеанса протокола L2TP. После завершения инициирования обоих подключений информация в виде кадров протокола PPP начинает передаваться по туннелю.

Формирование защищенного канала происходит в три этапа:

1) установление соединения клиента с сервером удаленного доступа;

2) аутентификация пользователя;

3) конфигурирование защищенного туннеля.

Для установления соединения с сервером удаленного доступа (сетевой сервер L2TP) удаленный пользователь связывается по протоколу PPP с концентратором доступа L2TP, обычно функционирующем на сервере провайдера. Концентратор доступа может выполнить аутентификацию пользователя от имени провайдера. По заданному имени получателя концентратор доступа определяет адрес сетевого сервера L2TP, который защищает сеть с заданным адресом. Между концентратором доступа и сервером L2TP устанавливается соединение. Далее производится аутентификация пользователя сервером L2TP. В случае успешной аутентификации устанавливается защищенный туннель между концентратором доступа и сервером L2TP. С помощью управляющих сообщений производится настройка параметров туннеля, причем в одном туннеле может быть несколько сеансов пользователя. При использовании IPSec пакеты L2TP инкапсулируются в UDP-пакеты, которые передаются концентратором доступа и сервером L2TP через IPSec-туннель (порт 1701/tcp).

6.3. Протокол IPSec

Несмотря на то что протокол IP стал наиболее используемым протоколом связи во всем мире и базовой технологией Интернета, он имеет множество существенных недостатков. Среди них необходимо отметить ограниченность адресного пространства, отсутствие автоматической конфигурации узлов сети и недостатки внутренней

безопасности. Главной причиной этих недостатков является то, что IP-протокол изначально не был предназначен для массового использования.

В качестве развития IP-протокола была разработана его новая версия IPv6 (IP-протокол версии 6), в которой пытались решить проблемы предшествующих версий. Поскольку принятие новой версии IP-протокола затруднительно, что вызвано постоянным ростом Интернета и громадным разнообразием установленной аппаратуры и программного обеспечения, то меры безопасности, включенные в IPv6, были перенесены в текущую версию IPv4 в виде дополнительного комплекта протоколов. Этот набор протоколов назвали комплектом протоколов IPSec (IPSec Protocol Suite).

Подключение по протоколу IPSec имеет два основных режима: транспортный (transport) и туннельный (tunnel).

Транспортный режим — это форма связи типа узел-узел, где применяется шифрование только содержательной части пакета. Из-за двухточечного характера связи соответствующее программное обеспечение необходимо загрузить (установить) на все связывающиеся между собой узлы сети, что представляет собой достаточно серьезную проблему. Этот режим VPN удобно использовать для зашифрованной связи между узлами одной сети.

Туннельный режим применяется при создании большинства VPN, потому что он шифрует весь оригинальный пакет. Режим туннелирования может применяться для организации связи типа узел — узел, узел — шлюз или шлюз — шлюз. При организации связи типа шлюз-шлюз значительно упрощается связь между сетями, не требуется установка специального ПО на узлах сети.

Первая цель семейства протоколов IPSec состоит в обеспечении конфиденциальности, целостности и аутентификации информации, передаваемой посредством IP-протокола. Это достигается с помощью протокола обмена интернет-ключами (Internet Key Exchange, IKE), протокола ESP и протокола AH. Комбинация этих трех протоколов обеспечивает безопасный обмен информацией.

Второй целью семейства протоколов IPSec является предоставление разработчикам ПО набора стандартов.

Установление безопасного соединения начинается с формирования *ассоциации обеспечения безопасности* (Security Association, SA) между двумя сообщающимися сторонами.

6.3.1. Ассоциация обеспечения безопасности

Основа ассоциации обеспечения безопасности заключается в соглашении двух сторон о том, как они могут безопасно передавать свою информацию. В процессе соглашения стороны оговаривают детали защищенного обмена. Результатом такого соглашения и яв-

ляется ассоциация обеспечения безопасности. Каждому сеансу связи сопоставляются две ассоциации — по одной на каждого партнера связи.

Положительными чертами протокола IPSec являются открытость его стандарта для поддержки множества протоколов и режимов связи, а также поддержка различных алгоритмов шифрования и различных хеш-функций.

Прежде чем договариваться об ассоциации обеспечения безопасности, необходимо локальное конфигурирование элементов протокола IPSec, которые данный партнер собирается поддерживать. Эти параметры настройки хранятся в базе данных политики безопасности (Security Policy Database, SPD).

После согласования ассоциация обеспечения безопасности содержится в базе данных ассоциации обеспечения безопасности (Security Association Database, SAD). Это необходимо, поскольку узел сети может инициализировать несколько сеансов, каждому из которых может соответствовать своя SA.

Поскольку для каждого сетевого устройства доступно множество сеансов протокола IPSec, для правильного функционирования процесса необходимо, чтобы каждый сеанс согласования SA имел свой собственный уникальный идентификатор. Этот идентификатор составляется из уникального индекса параметра обеспечения безопасности (Security Parameter Index, SPI), который определяет, какая запись БД SA соответствует рассматриваемому подключению. Кроме того, учитывается адрес назначения и идентификатор используемого протокола (ESP или AH). Приведем пример выборки из базы данных ассоциации обеспечения безопасности маршрутизатора Cisco, использующей протокол ESP:

```
inbound esp sas:
spi: 0x71BB425D (1908097629)
transform: esp-des esp-md5-hmac,
in use settings={ Tunnel,}
slot: 0, conn id: 2000, flow_id: 1, crypto map: mode
sa timing: remaining key lifetime (k/sec):
(4600800/3500)
IV size: 8 bytes
replay detection support: Y
```

Эта выборка содержит множество данных о конкретном подключении, например SPI-номер подключения, алгоритмы шифрования и вычисления значения хэш-функции, используемые для этого подключения, факт, что это подключение работает в туннельном режиме, и продолжительность существования такого подключения. Такая информация содержится в базе данных ассоциации обеспечения безопасности для каждого согласованного подключения.

6.3.2. Туннельный и транспортный режимы протокола IPSec

Подключение по протоколу IPSec имеет два основных режима: транспортный (transport) и туннельный (tunnel).

Транспортный режим — это форма связи типа узел — узел, где применяется шифрование только содержательной части пакета. Из-за двухточечного характера связи соответствующее ПО необходимо загрузить (установить) на все связывающиеся между собой узлы сети, что представляет собой достаточно серьезную проблему. Этот режим VPN удобно использовать для зашифрованной связи между узлами одной сети.

Туннельный режим применяется при создании большинства VPN, потому что он шифрует весь оригинальный пакет. Режим туннелирования может применяться для организации связи типа узел — узел, узел — шлюз или шлюз — шлюз. При организации связи типа шлюз — шлюз значительно упрощается связь между сетями, не требуется установка специального ПО на узлах сети.

6.3.3. Протокол обмена интернет-ключами

Протокол обмена интернет-ключами предназначен для аутентификации и согласования параметров обмена протокола IPSec. Протокол IKE представляет собой комбинацию двух протоколов: протокола управления ассоциациями и протокола управления ключами обеспечения безопасности в Интернете (Internet Security Association and Key Management Protocol, ISAKMP), называемых фазами установления. Управление ключами можно выполнять вручную или используя альтернативы протокола IKE, такие как безопасная служба доменных имен (Secure DNS), Photuris или простой протокол обмена интернет-ключами (Simple Key Internet Protocol, SKIP).

Первая фаза протокола IKE. На первой фазе протокола IKE удаленный пользователь начинает сеанс со шлюзовым устройством VPN. Первая фаза выполняет две функции: аутентификацию удаленного пользователя и обмен информацией об открытых ключах, которые будут использоваться во второй фазе.

Аутентификацию можно выполнить несколькими различными способами. Наиболее часто используются технология предварительно распространяемых ключей (pre-shared keys) и технология цифровых сертификатов. Термин «предварительно распространяемые ключи» означает, что значения ключей предварительно задаются на всех компьютерах, которые собираются устанавливать соединения через VPN (что является существенным недостатком). При втором способе используются цифровые удостоверения (цифровые сертификаты, digital certificates), которые могут назначаться отдельно для каждого

объекта, который соединяется с VPN. Цифровыми сертификатами можно удаленно управлять и администрировать из уполномоченного центра сертификации (Certificate Authority, CA).

Сертификационная служба является центральным элементом структуры, называемой инфраструктурой с открытым ключом (Public Key Infrastructure, PKI). За инфраструктурой PKI стоит концепция публично доступной структуры, распределяющей информацию об открытых (публичных) ключах.

В первой фазе при обмене аутентификационной информацией и параметрами безопасности могут использоваться два режима: основной (main mode) и агрессивный (aggressive mode). Различия между ними заключаются в количестве сетевых пакетов, которыми обмениваются стороны, и во времени, за которое генерируется открытый ключ.

Агрессивный режим использует дополнительный заголовок меньшего размера, но основной режим обладает большей безопасностью и используется чаще всего.

Вторая фаза протокола IKE. Во второй фазе протокола IKE согласовываются конкретные параметры ассоциации обеспечения безопасности IPSec. Данное согласование подобно агрессивному режиму обмена информацией первой фазы. После завершения второй фазы формируется SA и пользователь получает подключение к VPN.

Во второй фазе возможен единственный режим согласования — быстрый режим (quick mode). Быстрый режим представляет собой короткий обмен, использующий три пакета. Все обмены второй фазы зашифрованы с помощью согласованных во время первой фазы протоколов и типов кодирования. При этом используется только защита, основанная на использовании хеш-функции и нонсе (nonce), включаемых в сетевые пакеты для подтверждения их оригинальности (нонсе является подтверждением того факта, что информация о ключе исходит из ожидаемого источника. Нонсе — это случайное число, генерируемое инициатором связи, которое заверяется респондентом цифровой подписью и посылается обратно).

Данная реализация включает в себя также идентификатор поставщика (vendor ID. VID), позволяющий участникам межплатформенных взаимодействий делать предположения о возможностях и конфигурации их партнеров, которые могут иметь различных изготовителей.

После создания ассоциации обеспечения безопасности, используемой протоколом IKE, можно применять протоколы обеспечения безопасности. При построении VPN, основанной на протоколе IPSec, можно выбрать использование одного из протоколов (AH или ESP) или использовать их одновременно.

Управление ключами. Применяемый по умолчанию для IPSec протокол автоматизированного управления ключами называется ISAKMP/Oakley и состоит из следующих элементов:

• протокол определения ключей Oakley — протокол на основе алгоритма Диффи—Хеллмана, но обеспечивающий дополнительную защиту. Протокол Oakley называется общим, так как он не диктует использования каких-либо конкретных форматов;

• протокол защищенных связей и управления ключами в Интернете — обеспечивает основу схемы управления ключами и поддержку специального протокола и необходимых форматов процедуры согласования атрибутов защиты.

ISAKMP не заставляет использовать какой-то конкретный алгоритм обмена ключами, а предлагает использовать любой подобный алгоритм.

Протокол Oakley разработан в целях сохранения преимуществ алгоритма Диффи—Хеллмана и устранения его недостатков. Алгоритм Oakley характеризуется следующими особенностями:

1) использование механизмов рецептов (cookies) для защиты от атак засорения;

2) соглашение двух сторон о группе, которая определяет параметры алгоритма обмена ключами Диффи—Хеллмана;

3) использование оказий для противостояния атакам повтора сообщений;

4) обмен открытыми ключами Диффи—Хеллмана;

5) аутентификация обмена для противостояния атакам «человек посередине».

Рассмотрим эти особенности применительно к возможным атакам.

В случае атаки засорения атакующий фальсифицирует адрес законного источника и посылает жертве открытый ключ. Жертва производит значительный объем операций для вычисления секретного ключа. Многократно повторенные сообщения такого типа могут засорить атакуемую систему бесполезной работой.

Требование обмена рецептами означает, что каждая сторона в начальном сообщении должна послать псевдослучайное число (рецепт), которое другая сторона должна подтвердить. Это подтверждение должно повториться в первом сообщении обмена ключами. Если адрес источника был фальсифицирован атакующим, то он не получит ответа. Таким образом, атакующий может только заставить пользователя генерировать подтверждения, а не выполнять трудоемкие вычисления.

Рекомендуемый метод создания рецептов заключается в быстром вычислении хеш-функции (например, MD5) для адресов источника и назначения, портов UDP источника и назначения и локально генерируемого секретного значения.

Алгоритм Oakley поддерживает использование различных групп для обмена ключами. В каждой группе определяются два глобальных параметра и алгоритм. Имеющиеся спецификации определяют следующие группы.

Возведение в степень в арифметике классов вычетов с 768-битовым модулем:

$$q = 2^{768} - 2^{704} - 1 + 2^{64}([2^{638}\pi] + 149\,686),\ d = 2.$$

Возведение в степень в арифметике классов вычетов с 1024-битовым модулем:

$$q = 2^{1\,024} - 2^{960} - 1 + 2^{64}([2^{894}\pi] + 129\,093),\ \alpha = 2.$$

Возведение в степень в арифметике классов вычетов с 1 536-битным модулем:

$$2^{1\,536} - 2^{1\,472} - 1 + 2^{64}\,(2^{1\,406}\pi + 741\,804).$$

Группа эллиптической кривой над конечным полем из 2^{155} элементов: функция — $y^2 + xy = x^3 + ax + b$; генератор (в шестнадцатеричном виде): X = 7B, Y = 1C8; параметры эллиптической кривой (в шестнадцатеричном виде): $A = 0$, Y = 7 338F.

Группа эллиптической кривой над конечным полем из 2^{185} элементов:

функция — $y^2 + xy = x^3 + ax + b$; генератор (в шестнадцатеричном виде): X = 18, Y = D; параметры эллиптической кривой (в шестнадцатеричном виде): $A = 0$, Y = 1EE9.

Для защиты от атак воспроизведения сообщений применяются оказии. Каждая оказия представляет собой локально порожденное псевдослучайное число. Оказии появляются в ответах и шифруются на определенных стадиях обмена данными.

С алгоритмом Oakley могут применяться три различных метода аутентификации.

Цифровые подписи — аутентификация обмена данными осуществляется с помощью подписи доступной обеим сторонам хеш-функции: каждая сторона шифрует хеш-код своим секретным ключом. Хеш-код генерируется для отдельных важных параметров, например для идентификатора пользователя и оказии.

Шифрование с открытым ключом — аутентификация обмена данными осуществляется с помощью шифрования некоторых параметров обмена (например, идентификаторов и оказий) с использованием секретного ключа отправителя.

Шифрование с симметричным ключом — для идентификации обмена данными может использоваться шифрование параметров обмена по симметричной схеме с помощью ключа, получаемого с применением какого-то дополнительного механизма.

Спецификация Oakley включает в себя ряд примеров обмена, допустимых для данного протокола. Рассмотрим пример агрессивного обмена ключами (aggressive key exchange) между абонентами A и B, в котором требуется обмен только тремя сообщениями:

$A \rightarrow B$: CKY_A, OK_KEYX, GRP, g^x, EHAO, NIDP, ID_A, ID_B, N_A, S_{KA} [ID_A|| ID_B||N_A||GRP||g^x||EHAO];

B→A: CKY_B, CKY_A, OK_KEYX, GRP, g^y, EHAS, NIDP, ID_B, ID_A, N_B, S_{KB} [ID_B|| ID_A|| N_B|| N_A||GRP|| g^y ||EHAS];

A→B: CKY_A, CKY_B, OK_KEYX, GRP, g^x, EHAS, NIDP, ID_A, ID_B, N_A, N_B, S_{KA} [ID_A|| ID_B||N_A|| N_B|| GRP||g^x|| g^y ||EHAS].

Здесь CKY_A, CKY_B — рецепты абонентов A и B соответственно; OK_KEYX — тип сообщения обмена ключами; GRP — имя группы для этого обмена; g^x, g^y — открытые ключи абонентов; g^{xy} — сеансовый ключ для этого обмена; EHAO, EHAS — функции шифрования (E), хеширования (H), аутентификации (A), предложенные (O) и выбранные (S); NIDP — оставшаяся часть сообщения шифрованию не подлежит; ID_A, ID_B — идентификаторы абонентов; N_A, N_B — случайные числа абонентов для этого обмена; S_{KA} [...], S_{KB} [...] — подписи, использующие секретные ключи абонентов; || — символ конкатенации.

Протокол ISAKMP определяет процедуры и форматы пакета, используемые для переговоров о создании, изменении или удалении защищенных связей. Как часть процесса создания защищенной связи ISAKMP определяет полезную нагрузку сообщений обмена ключами и аутентификации данных. Сообщение ISAKMP состоит из заголовка и следующих за ним полезных нагрузок. Все это передается с помощью транспортного протокола (спецификации требуют, чтобы любая реализация поддерживала использование UDP). На рис. 6.3 показан формат заголовка ISAKMP, содержащего следующие поля:

- рецепт инициатора (64 бит);
- рецепт респондента (64 бит);
- следующая полезная нагрузка (8 бит) — указывает тип первой полезной нагрузки в сообщении;
- главный номер версии (4 бит) — указывает главный номер используемой версии ISAKMP;

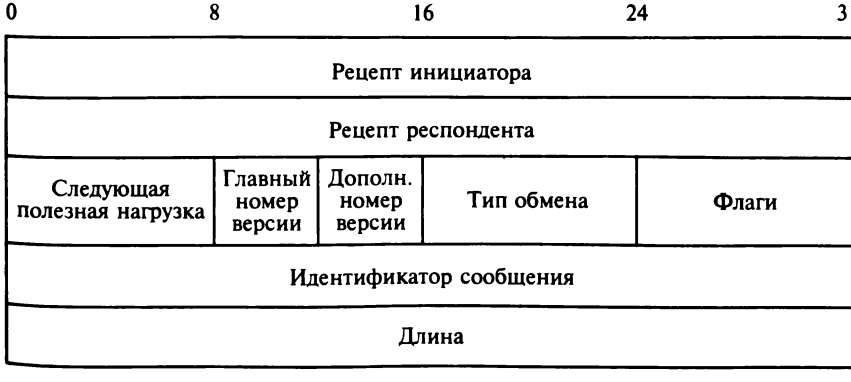


Рис. 6.3. Формат заголовка ISAKMP

• дополнительный номер версии (4 бит) — указывает дополнительный номер используемой версии;

• тип обмена (8 бит) — указывает тип обмена;

• флаги (8 бит) — указывают параметры, установленные для данного обмена ISAKMP. В настоящее время определено два бита: бит шифрования (устанавливается, когда все имеющиеся полезные нагрузки, следующие за заголовком, зашифрованы с использованием алгоритма шифрования этой защищенной связи) и бит фиксации (предназначен для гарантии того, что шифрованный материал не будет получен до начала защищенной связи);

• идентификатор сообщения (32 бит) — уникальный идентификатор данного сообщения;

• длина (32 бит) — длина всего сообщения (заголовка и всех полезных нагрузок в байтах).

Типы полезных нагрузок ISAKMP. Все полезные нагрузки ISAKMP имеют заголовки одного типа.

Формат заголовка типовой полезной нагрузки показан на рис. 6.4.

Поле следующей полезной нагрузки имеет значение 0, если данная полезная нагрузка является в сообщении последней (иначе значением поля будет тип следующей полезной нагрузки).

Значение поля длины полезной нагрузки указывает длину в байтах соответствующей полезной нагрузки, включая длину ее заголовка. В табл. 6.1 приведен список типов полезных нагрузок, допустимых для ISAKMP, и указаны поля (параметры), составляющие полезную нагрузку каждого из этих типов.

Полезный груз типа SA (защищенная связь) служит для того, чтобы начать процесс создания защищенной связи. В этом случае параметр Domain of Interpretation идентифицирует область интерпретации (DOI), в рамках которой выполняется согласование. Параметр Situation (ситуация) определяет политику защиты для процесса согласования, устанавливая степени защиты (например, гриф секретности, уровень безопасности).

Полезный груз типа P (предложение) включает в себя информацию, используемую в ходе согласования атрибутов создаваемой защищенной связи, а также указывает протокол для защищенной связи (ESP или AH), в отношении которой ведутся переговоры. Кроме того, полезная нагрузка этого типа включает в себя индекс параметров защиты объекта-отправителя и число трансформаций. Каждая

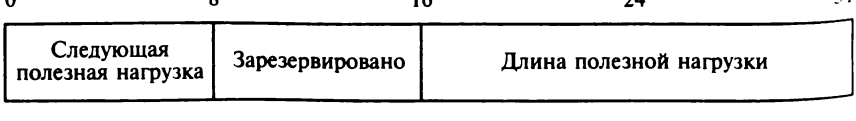


Рис. 6.4. Формат заголовка типовой полезной нагрузки

Таблица 6.1. **Типы полезных нагрузок ISAKMP**

Тип	Параметры	Описание
Защищенная связь (SA)	Область интерпретации, ситуация	Используется для согласования атрибутов защиты и указания области интерпретации и ситуации, в рамках которых выполняется такое согласование
Предложение (P)	Номер предложения, идентификатор протокола, длина индекса параметра защиты, число трансформаций, индекс параметров защиты	Используется в ходе согласования параметров создаваемой защищенной связи, указывает применяемый протокол и число трансформаций
Трансформация (T)	Номер трансформации, идентификатор трансформации, атрибуты защищенной связи	Применяется в ходе согласования параметров защищенной связи, указывает преобразование и соответствующие атрибуты защищенной связи
Обмен ключами (KE)	Данные обмена ключами	Поддерживает целый ряд методов обмена ключами
Идентификация (ID)	Тип идентификации, данные идентификации	Предназначен для обмена информацией идентификации
Сертификат (CERT)	Кодировка сертификата, данные сертификации	Служит для пересылки сертификатов и другой связанной с сертификатами информации
Запрос сертификата (CR)	Число типов сертификатов, типы сертификатов, число типов центров сертификации, центры сертификации	Используется для запросов сертификатов, указывает типы запрашиваемых сертификатов и приемлемые центры сертификации

Окончание табл. 6.1

Тип	Параметры	Описание
Хеширование (HASH)	Данные хеширования	Содержит данные, генерируемые функцией хеширования
Подпись (SIG)	Данные подписи	Содержит данные, генерируемые функцией цифровой подписи
Оказия (NONCE)	Данные оказии	Содержит оказию
Уведомление (N)	Область идентификации, идентификатор протокола, длина индекса параметров защиты, тип уведомления, индекс параметров защиты, данные уведомления	Используется для передачи данных уведомления, например признака возникновения ошибки
Удаление (D)	Область идентификации, идентификатор протокола, длина индекса параметров защиты, число индексов параметров защиты, индекс параметра защиты (один или несколько)	Указывает защищенную связь, которая больше не является действительной

трансформация содержится в полезной нагрузке типа T (трансформация).

Применение нескольких полезных нагрузок типа трансформации позволяет инициатору предложить на выбор несколько возможностей, из которых отвечающая сторона должна выбрать одну или отвергнуть предложение.

Полезный груз типа T (трансформация) определяет защитную информацию, которая должна использоваться для защиты коммуникационного канала в соответствии с указанным протоколом. Параметр Transform # (номер трансформации) позволяет идентифицировать данный полезный груз, чтобы отвечающая сторона могла сослаться на этот номер в подтверждение своего решения использовать именно

эту трансформацию. Поля Transform ID (идентификатор трансформации) и Attributes (атрибуты) определяют трансформацию (например, 3DES для ESP или HMAC-SHA-1—96 для AH) и связанные с ней параметры (например, длину хеш-кода).

Полезный груз типа KE (обмен ключами) может использоваться для целого ряда методов обмена ключами, среди которых Oakley, обмен по Диффи-Хеллману и обмен ключами RSA. Поле Key Exchange data (данные обмена ключами) содержит данные, необходимые для создания сеансового ключа и зависящие от используемого алгоритма обмена ключами.

Полезный груз типа ID (идентификация) предназначен для идентификации связывающихся сторон и может использоваться для проверки аутентичности информации. Как правило, поле ID data (данные идентификации) содержит адрес IPv4 или IPv6.

Полезный груз типа CERT (сертификат) несет сертификат открытого ключа. Поле Certificate Encoding (кодировка сертификата) указывает тип сертификата или связанную с сертификатом информацию, которая может обозначать следующее:

- сертификат X.509 в кодировке KCS #7;
- сертификат PGP;
- подписанный ключ DNS;
- сертификат X.509 — подпись;
- сертификат X.509 — обмен ключами;
- мандаты Kerberos;
- список отозванных сертификатов (CRL);
- список отозванных полномочий (ARL);
- сертификат SPKI.

Полезный груз типа CR (запрос сертификата) отправитель может разместить в любой точке обмена ISAKMP, чтобы запросить сертификат второй из участвующих в обмене сторон. Этот полезный груз может содержать список нескольких типов сертификатов и нескольких центров сертификации, которые рассматриваются как приемлемые.

Полезный груз типа HASH (хеширование) содержит данные, генерируемые функцией хэширования для некоторой части сообщения и (или) состояния ISAKMP. С помощью этого полезного груза можно проверить целостность данных в сообщении или аутентифицировать объекты, ведущие переговоры.

Полезный груз типа SIG (подпись) содержит данные, генерируемые функцией цифровой подписи для некоторой части сообщения и (или) состояния ISAKMP. Этот полезный груз служит для проверки целостности данных в сообщении и может использоваться в рамках сервиса, гарантирующего невозможность отказа от ответственности.

Полезный груз типа NONCE (оказия) включает в себя случайные данные, обеспечивающие гарантию выполнения процесса обмена в реальном времени и защиту его от атак воспроизведения сообщений.

Таблица 6.2. **Базовый обмен**

№ п/п	Сообщение обмена	Комментарий
1	A→B: SA, NONCE	Начинается согласование защищенной связи
2	B→A: SA, NONCE	Основные параметры защищенной связи согласованы
3	A→B: KE, ID_A, AUTH	Ключ сгенерирован, отвечающая сторона аутентифицировала инициатора
4	B→A: KE, ID_B, AUTH	Инициатор идентифицировал отвечающую сторону, ключ сгенерирован, защищенная связь установлена

Полезный груз типа N (уведомление) содержит информацию или об ошибке, или о состоянии, связываемом с данной защищенной связью и данным процессом согласования параметров защищенной связи.

Полезный груз типа D (удаление) указывает одну или несколько защищенных связей, которые предполагаются недействительными в виду того, что отправитель удалил их из своей базы данных.

Обмен ISAKMP. Протокол ISAKMP обеспечивает основу для обмена сообщениями, основными блоками которых являются полезные нагрузки. Спецификации указывают пять типов обмена, кото-

Таблица 6.3. **Обмен с защитой идентификации сторон**

№ п/п	Сообщение обмена	Комментарий
1	A→B: SA	Начинается согласование защищенной связи
2	B→A: SA	Основные параметры защищенной связи согласованы
3	A→B: KE, NONCE	Ключ сгенерирован
4	B→A: KE, NONCE	Ключ сгенерирован
5	B→A: ID_A, AUTH	Отвечающая сторона идентифицировала инициатора
6	A→B: ID_B, AUTH	Инициатор идентифицировал отвечающую сторону, защищенная связь установлена

Таблица 6.4. **Обмен только данными аутентификации**

№ п/п	Сообщение обмена	Комментарий
1	A→B: SA, NONCE	Начинается согласование защищенной связи
2	B→A: SA, NONCE ID_B, AUTH	Основные параметры защищенной связи согласованы, отвечающая сторона идентифицировала инициатора
3	A→B: ID_A, AUTH	Инициатор идентифицировал отвечающую сторону, защищенная связь установлена

рые должны поддерживаться по умолчанию: базовый обмен, обмен с защитой идентификации сторон, обмен только данными аутентификации, энергичный обмен и информационный обмен.

Базовый обмен позволяет провести обмен ключами и данными аутентификации одновременно (табл. 6.2). Это сводит к минимуму число сообщений обмена за счет отказа от защиты идентификации сторон. Два первых сообщения обеспечивают обмен рецептами и создают защищенную связь с согласованными протоколом и трансформациями. Обе стороны используют оказии, чтобы защититься от атак воспроизведения сообщений. В оставшихся двух сообщениях происходит обмен относящейся к ключам информацией и идентификаторами пользователей. Полезный груз AUTH используется для аутентификации ключей, участвующих в обмене сторон и оказий из первых двух сообщений.

Обмен с защитой идентификации сторон является расширением базового обмена в целях защиты информации об участвующих в

Таблица 6.5. **Энергичный обмен**

№ п/п	Сообщение обмена	Комментарий
1	A→B: SA, KE, NONCE, ID_A	Начинается согласование защищенной связи и обмен ключами
2	B→A: SA, KE, NONCE, ID_B, AUTH	Отвечающая сторона идентифицировала инициатора, ключ сгенерирован, основные параметры защищенной связи согласованы
3	A→B: AUTH	Ключ сгенерирован, отвечающая сторона аутентифицировала инициатора

Таблица 6.6. **Информационный обмен**

№ п/п	Сообщение обмена	Комментарий
1	A→B: N/D	Уведомление об ошибке или состоянии или уведомление об удалении связи

обмене сторонах (табл. 6.3). Два первых сообщения создают защищенную связь. Два следующих сообщения осуществляют обмен ключами и включают в себя две оказии для защиты от атак воспроизведения. Как только становится доступным сеансовый ключ, стороны обмениваются шифрованными сообщениями (в сообщениях 5 и 6 полезный груз после заголовка ISAKMP шифруется), содержащими информацию аутентификации, например цифровые подписи, и, если необходимо, сертификаты открытых ключей.

Обмен только данными аутентификации используется для того, чтобы осуществить взаимную аутентификацию без обмена ключами (табл. 6.4). Два первых сообщения создают защищенную связь. Кроме того, отвечающая сторона использует второе сообщение для того, чтобы передать свой идентификатор, и применяет аутентификацию для защиты сообщения. Инициатор посылает третье сообщение, чтобы передать свой аутентифицированный идентификатор.

Энергичный обмен сводит к минимуму число сообщений обмена за счет отказа от защиты идентификации сторон (табл. 6.5). В первом сообщении инициатор предлагает создать защищенную связь с набором предлагаемых протоколов и трансформаций. Кроме того, инициатор начинает обмен ключами и высылает свой идентификатор. Во втором сообщении отвечающая сторона указывает на принятие защищенной связи с уже конкретными протоколом и трансформацией, завершает обмен ключами и аутентифицирует переданную информацию. В третьем сообщении инициатор передает результат аутентификации полученной ранее информации, шифруя его с помощью общего секретного сеансового ключа.

Информационный обмен предназначен для односторонней пересылки информации, касающейся параметров управления защищенной связью (табл. 6.6). В этом сообщении полезный груз после заголовка ISAKMP шифруется.

6.3.4. Протокол аутентификации заголовка

Протокол AH — это IP-протокол 51. Он поддерживает функциональные возможности аутентификации и проверки целостности, но не поддерживает конфиденциальность содержимого пакета.

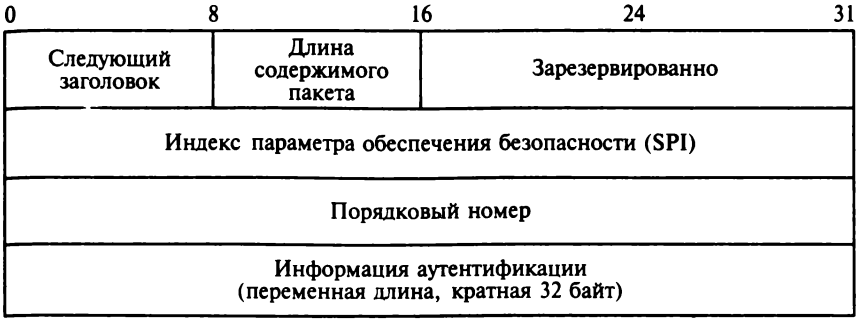


Рис. 6.5. Структура заголовка AH-пакета

Обеспечение аутентификации и защиты целостности достигается добавлением дополнительного заголовка к IP-пакету. Этот заголовок содержит цифровую подпись, называемую *значением проверки целостности* (Integrity Check Value, ICV), которая является в основном значением хеш-функции, подтверждающей, что пакет не был изменен во время транспорта.

Информация IP-протокола, содержащаяся в пакете, гарантирует правильность содержимого пакета, но она передается в открытом виде. Поскольку протокол AH просматривает заголовок IP-пакета при вычислении цифровой подписи, можно убедиться, что IP-адрес отправителя подлинный и что пакет исходит от того, кого требуется.

Протокол AH также поддерживает использование порядковых номеров (sequence numbers), помогающих предотвращать нападения,

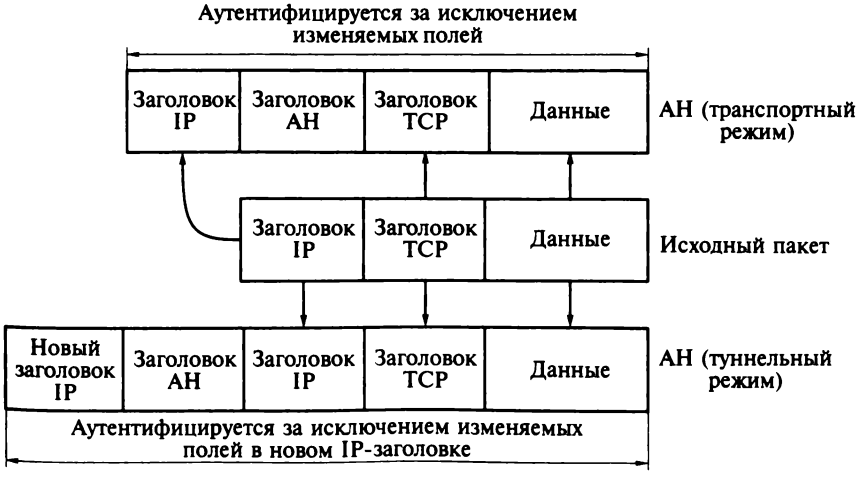


Рис. 6.6. Схема преобразования исходного пакета в AH-пакет

основанные на повторном использовании пакета (replay attack). Эти номера используются коммуникационными устройствами для отслеживания потока сетевых пакетов сеанса связи.

Использование информации IP-заголовка протоколом AH делает его несовместимым с использованием NAT. Структура заголовка AH-пакета представлена на рис. 6.5.

Поле следующего заголовка содержит идентификатор, определяющий тип заголовка пакета, следующего за заголовком AH-пакета.

Поле длины содержимого пакета определяет длину заголовка AH-пакета.

Индекс параметра обеспечения безопасности показывает, частью какого уникального потока связи ассоциации обеспечения безопасности является рассматриваемый пакет.

Порядковый номер — уникальное увеличивающееся значение, которое предназначено для противодействия повторному использованию пакета.

Поле информации аутентификации содержит значение проверки целостности и цифровую подпись, подтверждающую подлинность рассматриваемого пакета.

Схема преобразования исходного пакета в AH-пакет для различных режимов приведена на рис. 6.6.

6.3.5. Протокол безопасной инкапсуляции содержимого пакета

Протокол ESP — это IP-протокол 50. Он обеспечивает конфиденциальность при помощи полного шифрования содержимого IP-пакетов. Протокол ESP реализован в виде модулей и может использовать любое количество доступных симметричных алгоритмов шифрования, в числе которых DES, 3DES, IDEA. Применение протокола ESP различается в зависимости от используемого режима протокола IPSec.

В транспортном режиме протокол ESP просто добавляет свой собственный заголовок после IP-заголовка и зашифровывает остальную часть сетевого пакета начиная с транспортного уровня. Если при этом определена служба аутентификации, то протокол ESP добавляет концевую метку (trailer). Концевая метка предназначена для подтверждения целостности пакета и аутентификации (в отличие от протокола AH значение проверки целостности вычисляется без использования информации из IP-заголовка).

При использовании туннельного режима протокол ESP инкапсулирует оригинальный пакет полностью, шифруя его целиком и создавая новый IP-заголовок и ESP-заголовок в устройстве туннелирования. Концевая метка также добавляется в случае выбора аутентификационного сервиса протокола ESP.

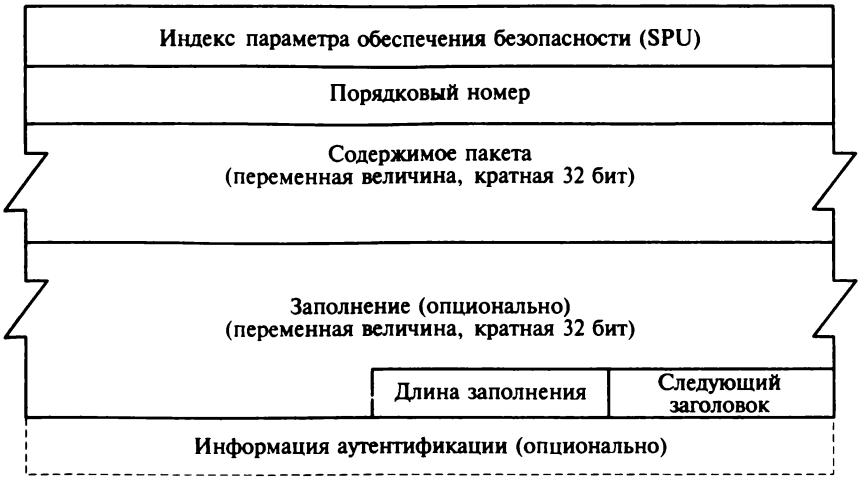


Рис. 6.7. Структура заголовка пакета ESP

В любом режиме протокол ESP использует в каждом сетевом пакете порядковые номера. При работе протокола ESP в туннельном режиме можно использовать NAT.

Структура заголовка пакета ESP приведена на рис. 6.7.

Поле длины заполнения указывает, насколько заполнено содержимое пакета (если такое вообще имеет место), чтобы длина содер-

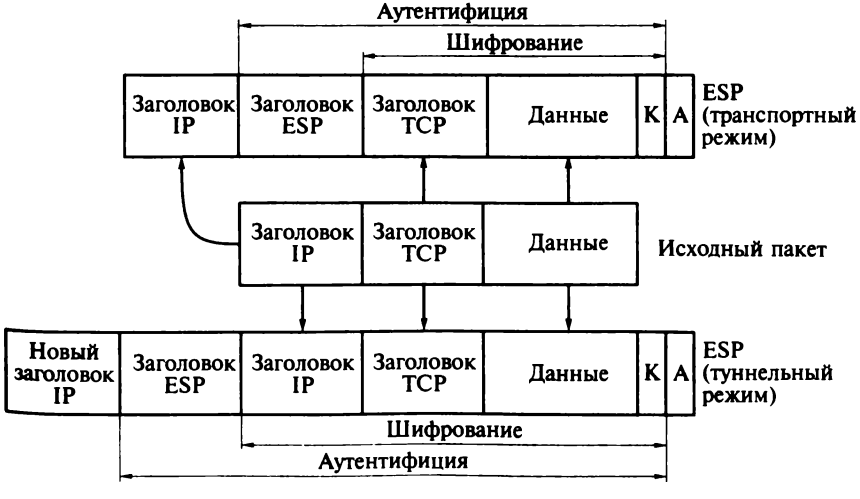


Рис. 6.8. Схема преобразования исходного пакета в пакет ESP

жимого пакета и последующие поля заголовка соответствовали требованию выравнивания длины сетевого пакета.

Поле следующего заголовка сообщает номер протокола пакета, который инкапсулирован внутри пакета протокола ESP.

Поле информации аутентификации содержит дополнительное значение проверки целостности, которое доступно для пакетов протокола ESP.

Схема преобразования исходного пакета в пакет ESP для различных режимов приведена на рис. 6.8.

6.3.6. Пример применения протокола IKE

Проиллюстрируем применение протокола примером для маршрутизатора Cisco, который можно сконфигурировать как часть VPN различными способами. В этом примере создается туннель от одного маршрутизатора к другому с применением протокола ESP и алгоритма SHA-1. В первом фрагменте устанавливаются параметры настройки управления ассоциациями и ключами обеспечения безопасности в Интернете для конфигурации протокола IKE следующим образом:

```
crypto isakmp policy 10
authentication pre-share
crypto isakmp key <s3cr3t!> address 192.168.100.100
```

В данном случае в качестве типа аутентификации определено использование предварительно распространяемого (*pre-share*) ключа, а в следующей строке задано само значение ключа *<s3cr3t!>* для заданного адреса 192.168.100.100. Далее задаются опции безопасности, которые этот маршрутизатор будет требовать при согласовании ассоциации обеспечения безопасности. Эти опции называются набором преобразования (*transform-set*):

```
crypto ipsec transform-set secure esp-3des esp-sha-
hmac
```

Набор преобразования *secure* устанавливается с перечислением допустимых сочетаний протоколов и алгоритмов аутентификации. Следующая команда используется для назначения интерфейсу соответствующей карты шифрования (*crypto map*), которой присвоено имя *cmap*:

```
crypto map cmap local-address Ethernet1
```

Карта шифрования представляет собой набор конфигурационной информации, определяющей трафик, который должен проходить по VPN:

```
crypto map cmap 10 ipsec-isakmp
set peer 192.168.200.200
set transform-set secure
match address 101
```

Эти команды связывают вместе все компоненты конфигурации VPN, они дают равноправный адрес (*peer address*) объекта, к которому будет создан туннель, список разрешенных наборов преобразования и определяют список контроля доступа для трафика, имеющего номер 101, который должен быть направлен через VPN (это расширенный список):

```
access-list 101 permit ip 192.168.2.0 0.0.0.255
192.168.1.0 0.0.0.255
access-list 101 deny ip 192.168.2.0 0.0.0.255 any
```

Этот список контроля доступа определяет трафик VPN, пакеты которого должны зашифровываться и направляться в соответствующую конечную точку VPN. Далее следует фрагмент конфигурации интерфейса маршрутизатора:

```
interface Ethernet1
description external interface
ip address x.x.x.x
crypto map cmap
```

Затем необходимо сконфигурировать службу трансляции адресов, чтобы позволить трафику защищенного туннеля обходить эту трансляцию (соответствующая маршрутная карта называется *nonat*):

```
ip nat inside source route-map nonat interface
Ethernet1 overload
route-map nonat permit 10
match ip address 102
```

Эта маршрутная карта допускает трафик, который запрещается в списке контроля доступа для обхода трансляции сетевых адресов, в то время как устройство трансляции разрешает трафик. Далее следуют правила списка контроля доступа, которые определены в маршрутной карте *nonat*:

```
access-list 102 deny ip 192.168.2.0 0.0.0.255
192.168.1.0 0.0.0.255
access-list 102 permit ip 192.168.2.0 0.0.0.255 any
```

Этот список используется для обхода трансляции сетевых адресов и разрешения трафика, связанного с протоколом IPSec, который инициализирует туннель:

```
access-list 112 permit udp any any eq isakmp
access-list 112 permit esp any any
```

Этот список контроля доступа сконфигурирован таким образом, чтобы разрешить трафики протокола ISAKMP (порт 500/udp) и ESP (IP-протокол 50. Протокол AH имеет номер 51).

6.3.7. Совместное использование протоколов ESP и AH

Отдельная защищенная связь может использовать либо протокол AH, либо протокол ESP, но никак не оба эти протокола одновременно. Тем не менее иногда конкретному потоку данных может требоваться и сервис AH, и сервис ESP.

Во всех случаях одному потоку для получения комплекса услуг IPSec требуется несколько защищенных связей. Такой набор защищенных связей называется пучком защищенных связей (security association bundle), посредством которого потоку должен предоставляться необходимый набор услуг IPSec. При этом защищенные связи в пучке могут завершаться в различных конечных точках.

Защищенные связи могут быть объединены в пучки следующими двумя способами:

1) транспортной смежности — в этом случае применяются несколько протоколов защиты к одному IP-пакету без создания туннеля. Эта комбинация AH и ESP эффективна только для одного уровня вложенности, так как обработка выполняется в одной точке — IPSec конечного получателя;

2) повторного туннелирования — в этом случае применяются несколько уровней протоколов защиты с помощью туннелирования IP. Этот подход допускает множество уровней вложения, поскольку туннели могут начинаться и завершаться в разных использующих IPSec узлах сети вдоль маршрута передачи данных.

Кроме того, эти два подхода можно объединить. Тогда при рассмотрении пучков защищенных связей возникает вопрос — в каком порядке могут применяться аутентификация и шифрование между данной парой конечных узлов. Рассмотрим несколько подходов к такому комбинированию.

В случае применения *ESP с опцией аутентификации* пользователь сначала применяет ESP к требующим защиты данным, а затем добавляет поле данных аутентификации. В зависимости от используемых режимов возможны следующие варианты (в обоих вариантах аутентификации подлежит шифрованный текст):

- транспортный режим ESP — аутентификация и шифрование применяются к полезному грузу IP, доставляемому узлу адресата, но при этом заголовок IP не защищается;
- туннельный режим ESP — аутентификация применяется ко всему пакету IP, доставляемому по адресу IP внешнего получателя (например, МЭ), и выполняется этим получателем. Весь внутренний пакет IP защищается механизмом секретности, поскольку предназначен для доставки внутреннему адресату IP.

В случае *транспортной смежности* используется пучок из двух защищенных связей в транспортном режиме, где внутренняя связь является защищенной связью ESP, а внешняя — защищенной связью

AH. В этом случае ESP используется без опции аутентификации. Поскольку внутренняя защищенная связь используется в транспортном режиме, шифрованию подлежит полезный груз IP. Получаемый в результате пакет состоит из заголовка IP, за которым следуют данные ESP. Затем применяется AH в транспортном режиме, так что аутентификация будет охватывать данные ESP и оригинальный заголовок IP, за исключением изменяемых полей. Достоинством данного подхода является то, что при комбинированном подходе аутентификация охватывает больше полей, включая поля адресов IP источника и назначения. Недостаток связан с использованием двух защищенных связей вместо одной.

Рассмотрим *транспортно-туннельный режим*. В некоторых случаях целесообразным является применение аутентификации до шифрования. Для этого используется пучок защищенных связей, состоящий из внутренней защищенной транспортной связи AH и внешней защищенной туннельной связи ESP. В этом случае аутентификация охватывает полезный груз IP вместе с заголовком IP (и расширениями), за исключением изменяемых полей. Полученный таким образом пакет затем обрабатывается в туннельном режиме ESP, в результате чего внутренний пакет вместе с данными аутентификации оказывается зашифрованным и имеющим новый внешний заголовок IP (с необходимыми расширениями).

6.3.8. Основные типы защищенных связей

В RFC 2401 приведены четыре примера комбинации защищенных связей, которые должны поддерживаться использующими IPSec узлами или шлюзами защиты. Рассмотрим эти примеры.

1. Обеспечение защиты связи между конечными системами, использующими IPSec (рис. 6.9). Эти конечные системы должны использовать общие секретные ключи. При этом допустимы следующие комбинации:
   - AH в транспортном режиме;
   - ESP в транспортном режиме;
   - сначала AH, а затем ESP в транспортном режиме;
   - любая из предыдущих связей внутри AH или ESP в туннельном режиме.
2. Обеспечение защиты между шлюзами (рис. 6.10). В этом случае защита организуется только между шлюзами (маршрутизаторами, МЭ), а в конечных узлах применение IPSec не предполагается. Требуется только одна туннельная защищенная связь. Туннель может использовать AH, ESP или ESP с опцией аутентификации.
3. Обеспечение сквозной защиты (рис. 6.11). В этом случае используется схема защиты между шлюзами с добавлением сквозной защиты. Туннель от шлюза к шлюзу обеспечивает аутентификацию

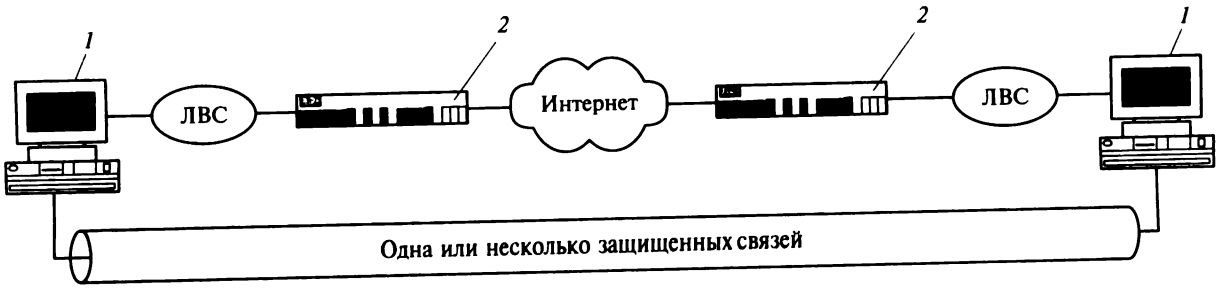


Рис. 6.9. Защищенная связь между конечными системами, использующими IPSec:
1 — пользователь; *2* — маршрутизатор

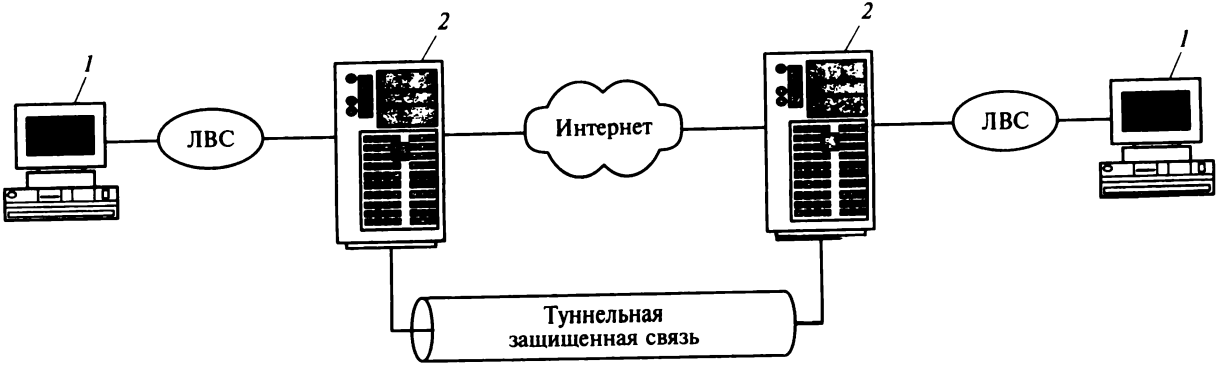


Рис. 6.10. Защищенная связь между шлюзами:
1 — пользователь; *2* — шлюз IPSec

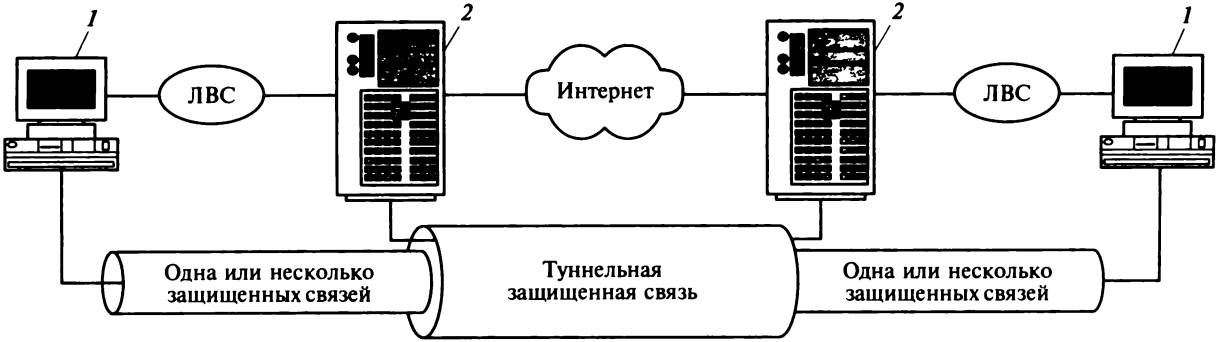


Рис. 6.11. Сквозная защита:
1 — пользователь; *2* — шлюз IPSec

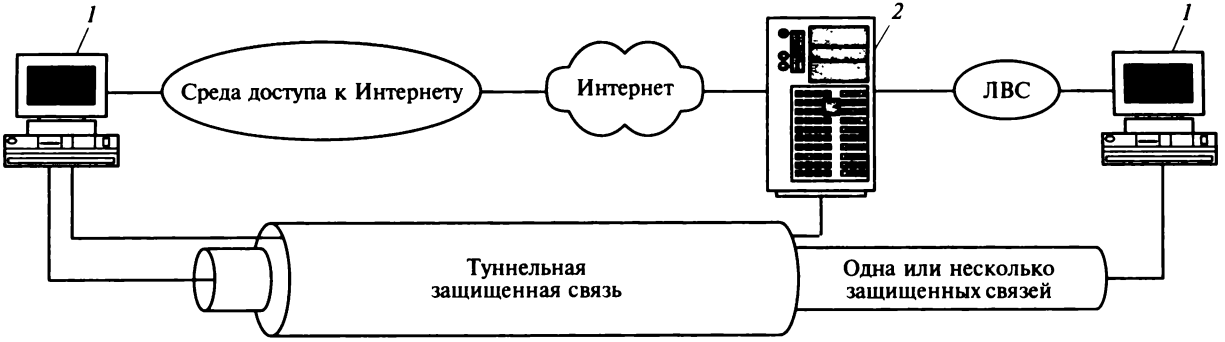


Рис. 6.12. Защита удаленного доступа:
1 — пользователь; *2* — шлюз IPSec

и (или) конфиденциальность для всего трафика между конечными системами (когда туннель использует ESP, обеспечивается ограниченная конфиденциальность потока обмена данными). Конечные системы могут использовать дополнительные сервисы IPSec, необходимые для конкретных узлов.

4. Обеспечение защиты удаленного доступа (рис. 6.12). В этом случае обеспечивается защита удаленного узла, использующего Интернет для связи с МЭ организации в целях получения доступа к узлу локальной вычислительной сети, защищаемому этим МЭ. Между удаленным узлом и МЭ необходимо использовать только туннельный режим. Как и в варианте 1, между удаленным и локальным узлами можно использовать одну или две защищенные связи.

6.4. Протоколы VPN транспортного уровня

Протоколы транспортного уровня прозрачны для прикладных протоколов и протоколов представления сервисов (HTTP, FTP, POP3, SMTP, NNTP и др.). Так как транспортный уровень отвечает за установку логических соединений и управление этими соединениями, то на этом уровне появляется возможность использования программ-посредников, которые проверяют допустимость соединений и обеспечивают выполнение других функций защиты.

Среди протоколов транспортного уровня наибольшее распространение получил протокол Secure Socket Layer/Transport Layer Security (SSL/TLS, разработанный компанией Netscape Communi-cations. Для выполнения функций посредничества между двумя взаимодействующими сторонами организацией IETF в качестве стандарта разработан протокол SOCKS.

Протокол SSL. Протокол SSL/TLS (SSL) изначально ориентировался на организацию защищенного обмена между клиентом и сервером. Клиентская часть протокола включена в состав всех популярных Web-браузеров, а серверная — в большинство Web-серверов как коммерческих, так и свободно распространяемых.

Протокол SSL устанавливает защищенные туннели между конечными точками (клиентом и сервером). Формирование защищенного обмена проходит в две стадии:

1) установление SSL-сеанса;

2) защищенное взаимодействие.

Процедура установления SSL-сеанса выполняется по протоколу Handshake Protocol, входящему в состав SSL. В процессе установления SSL-сеанса выполняются следующие основные действия:

- клиент посылает серверу запрос на установление защищенного соединения, в котором передаются параметры соединения: текущее время и дата, случайная последовательность RAND_CL, набор поддерживаемых клиентом алгоритмов сжатия, функций хэширования и криптографических алгоритмов;

• сервер, обработав запрос, передает клиенту согласованный набор параметров, в который входят: идентификатор сеанса, алгоритмы и функции из числа предложенных, сертификат сервера и случайная последовательность RAND_SERV;

• клиент проверяет сертификат сервера и при положительном результате проверки выполняет следующие операции: генерирует случайную последовательность (48 байт) PreMasterSecret, шифрует ее открытым ключом сервера и посылает серверу; с помощью согласованной хэш-функции формирует ключ MasterSecret, используя PreMasterSecret, RAND_CL и RAND_SERV; с помощью MasterSecret формирует сеансовые ключи;

• сервер расшифровывает последовательность PreMasterSecret и выполняет аналогичные клиенту операции;

• для проверки идентичности параметров SSL-сеанса клиент и сервер посылают друг другу текстовые сообщения, состоящие из данных, которыми клиент и сервер обменивались на предыдущих шагах. В случае успешной расшифровки и проверки целостности каждой из сторон считается, что сеанс установлен и стороны переходят в режим защищенного взаимодействия.

В ходе защищенного взаимодействия обе стороны при передаче формируют MAC для каждого сообщения и зашифровывают исходное сообщение MAC-кодом. При приеме сообщение расшифровывается и осуществляется проверка его целостности.

Особенностью протокола SSL является то, что при использовании SSL вне США действуют экспортные ограничения на длину ключей шифрования, такие же ограничения относятся и к алгоритмам шифрования Web-браузеров.

Протокол SOCKS. Протокол SOCKS разработан для организации функций посредничества между клиент-серверными приложениями, причем не привязан к протоколу IP и не зависит от ОС. Наибольшее распространение получили версии 4 и 5 этого протокола, причем SOCKS v5 одобрен IETF (Internet Engineering Task Force) в качестве стандарта. Протокол SOCKS v5 является расширением четвертой версии и реализует следующие дополнительные возможности:

• клиент может передавать серверу IP-адрес назначения или его DNS-имя;

• поддержка наравне с TCP и протокола UDP;

• при аутентификации пользователей сервер может согласовывать с клиентом способ аутентификации, допуская двустороннюю аутентификацию;

• возможность использования расширенной схемы адресации по IPv6.

Серверная часть обычно располагается на МЭ сети, а клиентская — на компьютере пользователя. Обобщенная схема установления соединения включает в себя следующие основные действия:

• пользователь выдает запрос, перехватываемый SOCKS-клиентом;

• SOCKS-клиент соединяется с SOCKS-сервером и сообщает ему идентификаторы поддерживаемых методов аутентификации;

• SOCKS-сервер выбирает метод аутентификации (при невозможности использовать предложенные методы соединение разрывается);

• в соответствии с выбранным методом SOCKS-сервер аутентифицирует пользователя (при неудаче аутентификации соединение разрывается);

• при успешной аутентификации SOCKS-клиент передает DNS-имя или IP-адрес запрашиваемого прикладного сервера;

• на основе имеющихся правил разграничения доступа SOCKS-сервер разрешает или запрещает установление соединения с этим прикладным сервером;

• при установленном соединении клиент и прикладной сервер взаимодействуют друг с другом, а SOCKS-клиент и SOCKS-сервер ретранслируют данные, выполняя функции посредников.

Аутентификация пользователей может основываться на паролях или цифровых сертификатах X.509. Для шифрации трафика могут использоваться любые протоколы. Популярные браузеры Netscape Navigator и Internet Explorer поддерживают протокол SOCKS.

6.5. Цифровые сертификаты

Для организации защищенных связей необходимо применение шифрования, которое, в свою очередь, требует наличия у пользователя ключа шифрования. При этом возникает проблема управления ключами, которая включает в себя следующие задачи: генерацию, проверку, распространение, использование, хранение, резервирование, обновление, уничтожение ключей и установление времени жизни ключа. При использовании симметричного шифрования у пользователя должны быть ключи для всех абонентов, с которыми он должен поддерживать защищенную связь, поэтому наибольшее распространение получило асимметричное шифрование. Однако из-за трудоемкости вычислений обычно асимметричное шифрование применяется для формирования сеансового ключа, который используется для симметричного шифрования данных в текущем сеансе.

Применение асимметричного шифрования требует наличия общедоступной системы, содержащей открытые ключи абонентов. В этом случае возможна фальсификация открытого ключа. Данная проблема решается с помощью сертификатов открытых ключей. Под сертификатом понимается подписанная цифровой подписью запись данных, содержащая имя и открытый ключ абонента. Сертификат подписывается специально создаваемой службой — уполномоченным центром сертификации, который пользуется доверием

абонентов. Универсальное распространение получила схема создания сертификатов открытых ключей, основанная на стандарте X.509. Формальное описание структуры сертификата выглядит следующим образом:

```
Certificate ::= SEQUENCE {
   tbsCertificate TBSCertificate,
   signatureAlgorithm AlgorithmIdentifier,
   signatureValue BIT STRING }
TBSCertificate ::= SEQUENCE {
   version [0] EXPLICIT Version DEFAULT v1,
   serialNumber CertificateSerialNumber,
   signature AlgorithmIdentifier,
   issuer Name,
   validity Validity,
   subject Name,
   subjectPublicKeyInfo SubjectPublicKeyInfo,
   issuerUniqueID [1] IMPLICIT UniqueIdentifier
   OPTIONAL,
            -- If present, version shall be v2 or
v3
   subjectUniqueID [2] IMPLICIT UniqueIdentifier
   OPTIONAL,
            -- If present, version shall be v2 or
v3
   extensions [3] EXPLICIT Extensions OPTIONAL
            -- If present, version shall be v3}

Version ::= INTEGER { v1(0), v2(1), v3(2) }

CertificateSerialNumber ::= INTEGER

Validity ::= SEQUENCE {
   notBefore Time,
   notAfter Time }

Time ::= CHOICE {

   utcTime UTCTime,
   generalTime GeneralizedTime }

UniqueIdentifier ::= BIT STRING

SubjectPublicKeyInfo ::= SEQUENCE {
   algorithm AlgorithmIdentifier,
   subjectPublicKey BIT STRING }

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

Extension ::= SEQUENCE {
   extnID OBJECT IDENTIFIER,
```

Версия
Номер сертификата
Идентификатор алгоритма шифрования и параметры алгоритма подписи
Имя объекта, выдавшего сертификат
Срок действия сертификата (не ранее $T_н$, не позднее $T_к$)
Имя субъекта
Открытый ключ субъекта, идентификатор алгоритма и параметры алгоритма
Идентификатор объекта, выдавшего сертификат
Идентификатор объекта, получившего сертификат
Расширения
Алгоритм подписи, параметры алгоритма, цифровая подпись сертификата

Рис. 6.13. Формат сертификата X.509

```
critical BOOLEAN DEFAULT FALSE,
extnValue OCTET STRING }
```

Каждый пользователь может доставить свой открытый ключ в уполномоченный центр некоторым защищенным образом и получить соответствующий сертификат. Затем он может опубликовать полученный сертификат. Сертификаты X.509 используются в большинстве приложений сетевой защиты, включая IPSec, SSL, SET, S/MIME и др. Формат сертификата, согласно X.509, приведен на рис. 6.13.

Кроме списка сертификатов открытых ключей абонентов уполномоченный центр содержит два важных списка, позволяющих выполнять функции управления сертификатами: список отозванных сертификатов (CRL), список отозванных полномочий (ARL).

Центр сертификации подписывает сертификат с помощью своего секретного ключа. Если все пользователи используют один центр сертификации, то это означает общее доверие данному центру. В X.509 предусмотрена иерархия центров, что позволяет создавать сложные системы управления ключами.

6.6. Примеры отечественного построения VPN

Протоколы построения защищенных виртуальных сетей могут быть реализованы различными средствами:

• серверами удаленного доступа, позволяющими создавать защищенные туннели на канальном уровне;

• маршрутизаторами, которые поддерживают протоколы создания защищенных виртуальных сетей на канальном и сетевом уровнях;

• межсетевыми экранами, возможно включающими в свой состав серверы удаленного доступа и позволяющие создавать VPN на канальном, сетевом и транспортном уровнях;

• специализированным программным обеспечением для создания защищенных туннелей на сетевом и транспортном уровнях;

• программно-аппаратными средствами, позволяющими формировать защищенные туннели на канальном и сетевом уровнях.

В качестве отечественного примера построения VPN рассмотрим особенности системы «Континент-К» НИП «Информзащита» и продукты ViPNet компании Infotecs.

Основными компонентами системы «Континент-К» являются криптошлюз, центр управления сетью криптошлюзов, программное обеспечение управления сетью криптошлюзов, абонентский пункт.

Криптошлюз представляет собой программно-аппаратное устройство под управлением операционной системы FreeBSD и обеспечивает:

• прием и передачу пакетов (TCP/IP);

• шифрование по ГОСТ 28147—89 в режиме гаммирования с обратной связью;

• сжатие защищаемых данных;

• защиту данных от искажения (имитовставка);

• NAT;

• аутентификацию удаленных абонентов;

• контроль целостности ПО криптошлюза до загрузки ОС (электронный замок «Соболь»).

Абонентский пункт служит для защищенного подключения к серверу доступа, на котором находится криптошлюз «Континент-К». После установления соединения и получения разрешения на доступ в защищаемую сеть (сервер доступа проводит идентификацию и аутентификацию удаленного пользователя и определяет его уровень доступа) весь трафик между абонентским пунктом и защищенной сетью шифруется по ГОСТ 28147—89. Комплекс использует схему сертификатов X.509.

Продукты ViPNet для организации VPN включают в себя два решения: ViPNet [Custom] и ViPNet [Tunnel].

Система ViPNet [Custom] предназначена для объединения в единую защищенную сеть произвольного числа рабочих станций и локальных сетей. В состав системы входят:

• ViPNet [Администратор] — модуль, создающий инфраструктуру сети, ведущий мониторинг сети и управляющий объектами сети. Он формирует первичную ключевую и парольную информацию для объектов сети и сертифицирует ключи, сформированные этими объектами;

• ViPNet [Координатор] — модуль, выполняющий маршрутизацию защищенных пакетов, туннелирование пакетов от обслуживаемой группы незащищенных компьютеров локальной сети. Он фильтрует трафик от источников, не входящих в VPN в соответствии с заданной ПБ, обеспечивает возможность работы защищенных компьютеров через межсетевые экраны и proxy-серверы других производителей;

• ViPNet [Клиент] — модуль, обеспечивающий защиту информации при передаче по открытым каналам связи и защиту доступа к ресурсам компьютера из сетей общего пользования. Модуль может быть установлен как на рабочую станцию, так и на сервер (баз данных, файл-сервер, www-сервер, SMTP, SQL и т.д.).

Система ViPNet [Tunnel] предназначена для объединения локальных сетей или офисов в VPN. Система базируется на пакетах программ ViPNet [Координатор], выполняющих роль шлюза, и клиентского программного обеспечения для компьютеров защищаемых локальных сетей. Система выполняет функции:

• сервера IP-адресов;

• proxy-сервера защищенных соединений;

• туннельного сервера;

• МЭ (фильтрация трафика от источников, не входящих в состав VPN, в соответствии с ПБ).

Во всех продуктах серии используется низкоуровневый драйвер, взаимодействующий непосредственно с драйвером сетевого интерфейса, что обеспечивает независимость от операционной системы. Криптографическим ядром данного семейства продуктов является средство криптографической защиты «Домен-К».

Рассмотренные отечественные продукты являются сертифицированными.

6.7. Инфраструктура PKI

Рассмотренные методы и средства формирования VPN-систем основываются на использовании технологии открытых ключей. При этом предполагалось, что пользователи VPN уже имеют секретные и открытые ключи, необходимые для выполнения различных криптографических операций. Одной из главных задач при формировании VPN является задача организации управления ключами. Решение этой задачи базируется на формировании специальной инфраструктуры открытых ключей (Public Key Infrastructure, PKI).

Под *инфраструктурой* понимается комплекс взаимосвязанных обслуживающих структур или объектов, составляющих и (или) обеспечивающих основу функционирования системы. Инфраструктура открытых ключей создается для организации управления открытыми ключами, освобождая участников криптосистемы от необходимости

решения второстепенных для них задач. *Инфраструктура открытых ключей* — это набор средств (технических, материальных, людских и т.д.), распределенных служб и компонентов, в совокупности используемых для поддержки криптографических задач на основе использования закрытого и открытого ключей. Она предназначена для обеспечения (с помощью сертификатов ключей) доверия законных пользователей к подлинности ключей, соответствия ключей пользователям и оговоренным условиям их применения. При этом сертификатом ключа является структура данных заранее определенного формата, которая включает в себя открытый ключ, идентификационную информацию владельца ключа, а также другую служебную информацию (время действия и предназначение ключа, тип используемых криптографических алгоритмов и др.), заверенную электронной подписью уполномоченного лица доверенного центра сертификации (удостоверяющего центра).

Основными задачами PKI являются:

- поддержка жизненного цикла цифровых ключей и сертификатов (т. е. генерация ключей, создание и подпись сертификатов, их распределение и пр.);
- регистрация фактов компрометации ключей и публикация «черных» списков отозванных или аннулированных сертификатов;
- поддержка процессов идентификации и аутентификации пользователей;
- реализация механизма интеграции (основанного на PKI) существующих приложений и всех компонентов подсистемы безопасности.

В самой инфраструктуре открытых ключей выделяют логическую и физическую структуры. *Логическая структура* включает в себя механизмы, субъекты системы, правила и взаимосвязи, необходимые для доступа к криптографическим ключам и ассоциирования криптографических ключей с их владельцами. *Физическая структура* содержит программы, форматы данных, коммуникационные протоколы, политики и процедуры, необходимые для использования в рамках данной инфраструктуры. Инфраструктура открытых ключей создается для обеспечения всего жизненного цикла использования криптографических ключей.

Цель инфраструктуры открытых ключей состоит в управлении ключами и сертификатами, посредством которого организации и отдельные пользователи могут поддерживать надежную сетевую среду. Эта инфраструктура позволяет использовать сервисы аутентификации, шифрования и выработки цифровой подписи согласованно с широким кругом приложений, функционирующих в среде открытых ключей. Основными компонентами инфраструктуры являются:

- удостоверяющий центр (УЦ);
- регистрационный центр;

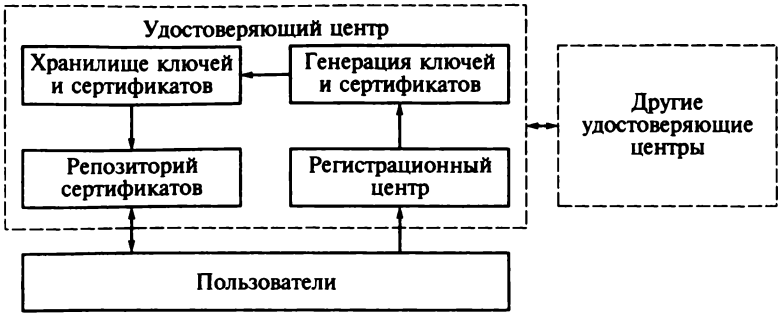


Рис. 6.14. Структурная схема инфраструктуры PKI

- сертификаты и репозиторий сертификатов;
- архив сертификатов;
- конечные пользователи (субъекты) системы (участники электронного взаимодействия — осуществляющие обмен информацией в электронной форме государственные органы, органы местного самоуправления, организации, а также граждане).

Структурная схема инфраструктуры открытых ключей приведена на рис. 6.14.

Кратко рассмотрим назначение основных компонентов инфраструктуры.

Для сообщества потенциальных пользователей наиболее практичным способом связывания открытых ключей и их владельцев является организация доверенных центров. Этим центрам участники электронного взаимодействия доверяют выполнение функций связывания ключей и идентификационных данных (идентичности) пользователей. Такие доверенные центры в терминологии PKI называются *удостоверяющими*; они сертифицируют связывание пары ключей с идентичностью, заверяя цифровой подписью структуру данных, которая содержит некоторое представление идентичности и соответствующего открытого ключа. Эта структура данных называется *сертификатом открытого ключа* (или просто сертификатом).

По сути, сертификат представляет собой некое зарегистрированное удостоверение, которое хранится в цифровом формате и признается сообществом пользователей PKI законным и надежным. Для заверения электронного сертификата используется электронная цифровая подпись УЦ — в этом смысле удостоверяющий центр уподобляется нотариальной конторе, так как подтверждает подлинность сторон, участвующих в обмене электронными сообщениями или документами. Пользователи могут легко идентифицировать сертификаты по имени УЦ и убедиться в их подлинности, используя его открытый ключ.

Регистрационный центр является необязательным компонентом инфраструктуры PKI. Обычно регистрационный центр получает от удостоверяющего центра полномочия регистрировать пользователей, обеспечивать их взаимодействие с УЦ и проверять информацию, которая заносится в сертификат.

Репозиторий — специальный объект инфраструктуры открытых ключей, содержащий базу данных, в которой хранится реестр сертификатов. Репозиторий предоставляет информацию о статусе сертификатов, обеспечивает хранение и распространение сертификатов, ведет списки аннулированных и отозванных сертификатов, управляет внесениями изменений в сертификаты.

На *архив сертификатов* возлагается функция долговременного хранения и защиты данных обо всех изданных сертификатах. Архив поддерживает базу данных, используемую при возникновении споров по поводу надежности электронных цифровых подписей, которыми в прошлом заверялись документы.

Архив подтверждает качество информации в момент ее получения и обеспечивает целостность данных во время хранения. Архив должен быть защищен соответствующими техническими средствами и процедурами.

Необходимо отметить, что эти перечисленные компоненты могут входить в состав самого удостоверяющего центра.

Основными сервисами инфраструктуры PKI являются криптографические сервисы и сервисы управления сертификатами.

К криптографическим сервисам относятся:

- *генерация пар ключей*. С помощью этого сервиса генерируется пара ключей (открытый ключ/секретный ключ). Секретный ключ хранится в файле, защищенном паролем или иными средствами (например, на смарт-карте или с помощью другого аппаратного или программного средства, гарантирующего конфиденциальность секретного ключа).

Для каждого пользователя генерируются две пары ключей (пользователь должен иметь одну пару ключей для зашифрования и расшифрования документов или сообщений, а другую пару — для выработки или проверки цифровой подписи);

- *выработка цифровой подписи*. Этот сервис заключается в генерации значения хеш-функции сообщения и его подписи;
- *верификация (проверка) цифровой подписи*. Посредством этого сервиса устанавливается подлинность сообщения и соответствующей ему цифровой подписи.

Сервисы управления сертификатами включают в себя:

- *выпуск сертификатов*. Сертификаты выпускаются УЦ для пользователей (физических и юридических лиц), для подчиненных ему удостоверяющих центров, а также для удостоверяющих центров других инфраструктур сторонних PKI;

• *управление жизненным циклом сертификатов и ключей.* Если секретный ключ пользователя потерян, похищен или скомпрометирован, либо существует вероятность наступления таких событий, действие сертификата должно быть прекращено. После получения подтверждения запроса пользователя об аннулировании сертификата УЦ уведомляет об аннулировании все заинтересованные стороны, формируя список аннулированных сертификатов;

• *поддержку репозитория.* Выпущенный сертификат включается в репозиторий (в соответствии со спецификациями стандарта X.500 или иными требованиями), чтобы третья сторона могла иметь к нему доступ. Кроме того, репозиторий содержит списки аннулированных сертификатов;

• *хранение сертификатов и списков аннулированных сертификатов.* Выпускаемые сертификаты и списки аннулированных сертификатов хранятся в архиве длительное время, которое определяется правилами хранения документов, заверенных электронной подписью.

Кроме этих основных функций инфраструктура PKI обеспечивает ряд важных вспомогательных функций: регистрацию пользователей (регистрацию и контроль информации о субъектах, а также аутентификацию субъектов); хранение информации в архиве; резервное хранение и восстановление ключей (УЦ должен иметь возможность восстановить зашифрованную информацию в случае потери пользователями их ключей шифрования).

В настоящее время основные положения инфраструктуры PKI закреплены в Федеральном законе от 6 апреля 2011 г. № 63-ФЗ «Об электронной подписи» (далее — Закон об электронной подписи).

Согласно ст. 5 Закона об электронной подписи установлены следующие виды электронной подписи.

1. Видами электронных подписей, отношения в области использования которых регулируются Законом об электронной подписи, являются простая электронная подпись и усиленная электронная подпись. Различаются усиленная неквалифицированная электронная подпись (далее — неквалифицированная электронная подпись) и усиленная квалифицированная электронная подпись (далее — квалифицированная электронная подпись).

2. Простой электронной подписью является электронная подпись, которая посредством использования кодов, паролей или иных средств подтверждает факт формирования электронной подписи определенным лицом.

3. Неквалифицированной электронной подписью является электронная подпись, которая:

1) получена в результате криптографического преобразования информации с использованием ключа электронной подписи;

2) позволяет определить лицо, подписавшее электронный документ;

3) позволяет обнаружить факт внесения изменений в электронный документ после момента его подписания;

4) создается с использованием средств электронной подписи.

4. Квалифицированной электронной подписью является электронная подпись, которая соответствует всем признакам неквалифицированной электронной подписи и следующим дополнительным признакам:

1) ключ проверки электронной подписи указан в квалифицированном сертификате;

2) для создания и проверки электронной подписи используются средства электронной подписи, получившие подтверждение соответствия требованиям, установленным в соответствии с Законом об электронной подписи.

5. При использовании неквалифицированной электронной подписи сертификат ключа проверки электронной подписи может не создаваться, если соответствие электронной подписи признакам неквалифицированной электронной подписи, установленным Законом об электронной подписи, может быть обеспечено без использования сертификата ключа проверки электронной подписи.

В Законе об электронной подписи определяется, что удостоверяющий центр — юридическое лицо или индивидуальный предприниматель, осуществляющие функции по созданию и выдаче сертификатов ключей проверки электронных подписей, а также иные функции, предусмотренные Законом об электронной подписи. Согласно ст. 13 Закона об электронной подписи удостоверяющий центр:

1) создает сертификаты ключей проверки электронных подписей и выдает такие сертификаты лицам, обратившимся за их получением (заявителям);

2) устанавливает сроки действия сертификатов ключей проверки электронных подписей;

3) аннулирует выданные этим удостоверяющим центром сертификаты ключей проверки электронных подписей;

4) выдает по обращению заявителя средства электронной подписи, содержащие ключ электронной подписи и ключ проверки электронной подписи (в том числе созданные удостоверяющим центром) или обеспечивающие возможность создания ключа электронной подписи и ключа проверки электронной подписи заявителем;

5) ведет реестр выданных и аннулированных этим удостоверяющим центром сертификатов ключей проверки электронных подписей (далее — реестр сертификатов), в том числе включающий в себя информацию, содержащуюся в выданных этим удостоверяющим центром сертификатах ключей проверки электронных подписей, и

информацию о датах прекращения действия или аннулирования сертификатов ключей проверки электронных подписей и об основаниях таких прекращения или аннулирования;

6) устанавливает порядок ведения реестра сертификатов, не являющихся квалифицированными, и порядок доступа к нему, а также обеспечивает доступ лиц к информации, содержащейся в реестре сертификатов, в том числе с использованием информационно-телекоммуникационной сети «Интернет»;

7) создает по обращениям заявителей ключи электронных подписей и ключи проверки электронных подписей;

8) проверяет уникальность ключей проверки электронных подписей в реестре сертификатов;

9) осуществляет по обращениям участников электронного взаимодействия проверку электронных подписей;

10) осуществляет иную связанную с использованием электронной подписи деятельность.

Инфраструктура открытых ключей предназначена для надежного функционирования криптографических информационных систем и позволяет как внутренним, так и внешним пользователям безопасно обмениваться информацией с помощью цепочки доверительных отношений. Решения на основе PKI способны обеспечить выполнение следующих требований по безопасности: обеспечение конфиденциальности, целостности, аутентичности, доступности и не отрицания авторства (неотказуемости).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие уровни стека протоколов TCP/IP используются для создания VPN?
2. Назовите основные типы конфигураций VPN.
3. Что понимается под термином «туннелирование»?
4. Назовите основные недостатки применения VPN.
5. Назовите основные этапы формирования защищенного канала при использовании протокола L2TP.
6. Назовите основные причины разработки IP-протокола версии 6.
7. Перечислите основные задачи, решаемые ассоциациями безопасности.
8. Перечислите основные различия транспортного и туннельного режимов.
9. Назовите основные элементы инфраструктуры PKI.
10. Оцените порядок «накладных расходов» при применении протоколов ESP и AH.
11. Возможно ли одновременное применение протоколов ESP и AH для организации защищенной связи?

12. Какие варианты защищенной связи поддерживаются протоколом IPSec?
13. Какой из протоколов (ESP или AH) несовместим с NAT?
14. Каковы состав и назначение цифрового сертификата открытых ключей?
15. Для чего в уполномоченном центре ведутся списки отозванных сертификатов и отозванных полномочий?
16. Какие средства могут быть использованы для построения VPN?

ПРИЛОЖЕНИЕ 1

Список основных терминов

Термин	Содержание (ссылка)
Активы (assets)	Все, что имеет ценность для организации (ГОСТ Р ИСО/МЭК 13335-1 — 2006). Информация или ресурсы, которые должны быть защищены средствами объекта оценки (ГОСТ Р ИСО/МЭК 15408-1 — 2008)
Анализ информационного риска	Систематическое использование информации для выявления угроз безопасности информации, уязвимостей информационной системы и количественной оценки вероятностей реализации угроз с использованием уязвимостей и последствий реализации угроз для информации и информационной системы, предназначенной для обработки этой информации (ГОСТ Р 50922 — 2006)
Анализ риска (risk analysis)	Систематический процесс определения величины риска (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Аудиторская проверка информационной безопасности в организации; аудит информационной безопасности в организации	Периодический независимый и документированный процесс получения свидетельств аудита и объективной оценки с целью определить степень выполнения в организации установленных требований по обеспечению информационной безопасности. Примечание. Аудит информационной безопасности в организации может осуществляться независимой организацией (третьей стороной) по договору с проверяемой организацией, а также подразделением или должностным лицом организации (внутренний аудит) (ГОСТ Р 50922 — 2006)
Аутентификация (подлинности субъекта доступа) (authentication))	Действия по проверке подлинности субъекта доступа в информационной системе (Р 50.1.056 — 2005)

Термин	Содержание (ссылка)
Аутентичность (authenticity)	Свойство, гарантирующее, что субъект или ресурс идентичны заявленным организации. Примечание. Аутентичность применяется к таким субъектам, как пользователи, к процессам, системам и информации (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Базовые защитные меры (baselinecontrols)	Минимальный набор защитных мер, установленный для системы или организации (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Безопасность информации (данных) (information (data) security)	Состояние защищенности информации (данных), при котором обеспечиваются ее (их) конфиденциальность, доступность и целостность. Примечание. Безопасность информации (данных) определяется отсутствием недопустимого риска, связанного с утечкой информации по техническим каналам, с несанкционированными и непреднамеренными воздействиями на данные и (или) на другие ресурсы автоматизированной информационной системы, используемые при применении информационной технологии (ГОСТ Р 50.1.056 — 2005)
Безопасность информационной технологии (IT security)	Состояние защищенности информационной технологии, при котором обеспечивается выполнение изделием, реализующим информационную технологию, предписанных функций без нарушений безопасности обрабатываемой информации (Р 50.1.056 — 2005)
Безопасность информационно-телекоммуникационных технологий (безопасность ИТТ) (ICT security)	Все аспекты, связанные с определением, достижением и поддержанием конфиденциальности, целостности, доступности, неотказуемости, подотчетности, аутентичности и достоверности информационно-телекоммуникационных технологий (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Воздействие (impact)	Результат нежелательного инцидента информационной безопасности (ГОСТ Р ИСО/МЭК 13335-1 — 2006)

Термин	Содержание (ссылка)
Вредоносная программа	Программа, используемая для осуществления несанкционированного доступа к информации и (или) воздействия на информацию или ресурсы автоматизированной информационной системы (ГОСТ Р 51275 — 2006)
Вредоносная программа	Программа, предназначенная для осуществления несанкционированного доступа к информации и (или) воздействия на информацию или ресурсы информационной системы (ГОСТ Р 50922 — 2006)
Достоверность (reliability)	Свойство соответствия предусмотренному поведению и результатам (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Доступность (availability)	Свойство объекта находиться в состоянии готовности и используемости по запросу авторизованного логического объекта (ГОСТ Р ИСО/МЭК 7498 — 99)
Доступность информации (ресурсов информационной системы) (availability)	Состояние информации (ресурсов информационной системы), при котором субъекты, имеющие право доступа, могут реализовать их беспрепятственно. Примечание. К правам доступа относятся: право на чтение, изменение, копирование, уничтожение информации, а также право на изменение, использование, уничтожение ресурсов (Р 50.1.056 — 2005)
Закладочное устройство; закладка	Элемент средства съема информации или воздействия на нее, скрытно внедряемый (закладываемый или вносимый) в места возможного съема информации. Примечание. Местами возможного съема информации могут быть ограждение, конструкция здания, оборудование, предметы интерьера, транспортные средства, а также технические средства и системы обработки информации (Р 50.1.056 — 2005)

Термин	Содержание (ссылка)
Защита информации; ЗИ	Деятельность, направленная на предотвращение утечки защищаемой информации, несанкционированных и непреднамеренных воздействий на защищаемую информацию (ГОСТ Р 50922—2006)
Защита информации от непреднамеренного воздействия	Защита информации, направленная на предотвращение воздействия на защищаемую информацию ошибок ее пользователя, сбоя технических и программных средств информационных систем, природных явлений или иных целенаправленных на изменение информации событий, приводящих к искажению, уничтожению, копированию, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации (ГОСТ Р 50922—2006)
Защита информации от несанкционированного воздействия; ЗИ от НСВ	Защита информации, направленная на предотвращение несанкционированного доступа и воздействия на защищаемую информацию с нарушением установленных прав и (или) правил на изменение информации, приводящих к разрушению, уничтожению, искажению, сбою в работе, незаконному перехвату и копированию, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации (ГОСТ Р 50922—2006)
Защита информации от несанкционированного доступа; ЗИ от НСД	Защита информации, направленная на предотвращение получения защищаемой информации заинтересованными субъектами с нарушением установленных нормативными и правовыми документами (актами) или обладателями информации прав или правил разграничения доступа к защищаемой информации. Примечание. Заинтересованными субъектами, осуществляющими несанкционированный доступ к защищаемой информации, могут быть: государство, юридическое лицо, группа физических лиц, в том числе общественная организация, отдельное физическое лицо (ГОСТ Р 50922—2006)

Термин	Содержание (ссылка)
Защитная мера (safeguard)	Сложившаяся практика, процедура или механизм обработки риска. Примечание. Следует заметить, что понятие «защитная мера» может считаться синонимом понятию «контроль» (ГОСТ Р ИСО/МЭК 13335-1 — 2006). Мера, используемая для уменьшения риска (ГОСТ Р 51898 — 2002)
Защищаемая информация	Информация, являющаяся предметом собственности и подлежащая защите в соответствии с требованиями правовых документов или требованиями, устанавливаемыми собственником информации. Примечание. Собственниками информации могут быть: государство, юридическое лицо, группа физических лиц, отдельное физическое лицо (ГОСТ Р 50922 — 2006)
Идентификация (identification)	Действия по присвоению субъектам и объектам доступа идентификаторов и (или) действия по сравнению предъявляемого идентификатора с перечнем присвоенных идентификаторов (Р 50.1.056 — 2005)
Информационная безопасность (information security)	Все аспекты, связанные с определением, достижением и поддержанием конфиденциальности, целостности, доступности, неотказуемости, подотчетности, аутентичности и достоверности информации или средств ее обработки (ГОСТ Р ИСО/МЭК 13335-1 — 2006). Свойство информации сохранять конфиденциальность, целостность и доступность. Примечание. Кроме того, данное понятие может включать в себя также и свойство сохранять аутентичность, подотчетность, неотказуемость и надежность (ГОСТ Р ИСО/МЭК 17999 — 2005)

Термин	Содержание (ссылка)
Инцидент информационной безопасности (information security incident)	Любое непредвиденное или нежелательное событие, которое может нарушить деятельность или информационную безопасность. Примечание. Инцидентами информационной безопасности являются: • утрата услуг, оборудования или устройств; • системные сбои или перегрузки; • ошибки пользователей; • несоблюдение политик или рекомендаций; • нарушение физических мер защиты; • неконтролируемые изменения систем; • сбои программного обеспечения и отказы технических средств; • нарушение правил доступа (ГОСТ Р ИСО/МЭК 13335-1 — 2006). Появление одного или нескольких нежелательных или неожиданных событий информационной безопасности, с которыми связана значительная вероятность компрометации бизнес-операций и создания угрозы информационной безопасности (ГОСТ Р ИСО/МЭК 18044 — 2007)
Источник угрозы информационной безопасности	Субъект (физическое лицо, материальный объект или физическое явление), являющийся непосредственной причиной возникновения угрозы безопасности информации (Р 50.1.056 — 2005)
Компьютерная атака (attack)	Целенаправленное несанкционированное воздействие на информацию, на ресурс автоматизированной информационной системы или получение несанкционированного доступа к ним с применением программных или программно-аппаратных средств (ГОСТ Р 51275 — 2006)
Компьютерный вирус	Программа, способная создавать свои копии (необязательно совпадающие с оригиналом) и внедрять их в файлы, системные области компьютера, компьютерных сетей, а также осуществлять иные деструктивные действия. При этом копии сохраняют способность дальнейшего распространения. Компьютерный вирус относится к вредоносным программам (ГОСТ Р 51188 — 98)

Термин	Содержание (ссылка)
Компьютерный вирус (computer virus)	Исполняемый программный код или интерпретируемый набор инструкций, обладающий свойствами несанкционированного распространения и самовоспроизведения. Примечание. Созданные дупликаты компьютерного вируса не всегда совпадают с оригиналом, но сохраняют способность к дальнейшему распространению и самовоспроизведению (Р 50.1.056 — 2005)
Компьютерный вирус	Вредоносная программа, способная создавать свои копии и (или) другие вредоносные программы (ГОСТ Р 51275 — 2006)
Контроль доступа (в информационной системе) (access control)	Проверка выполнения субъектами доступа установленных правил разграничения доступа в информационной системе (Р 50.1.056 — 2005)
Конфиденциальность (confidentiality)	Свойство информации быть недоступной и закрытой для неавторизованного индивидуума, логического объекта или процесса (ГОСТ Р ИСО 7498 — 1999). Состояние информации, при котором доступ к ней осуществляют только те субъекты, имеющие на него право (Р 50.1.056 — 2005)
Межсетевой экран	Локальное (однокомпонентное) или функционально-распределенное программное (программно-аппаратное) средство (комплекс), реализующее контроль за информацией, поступающей в автоматизированную систему и (или) выходящей из автоматизированной системы (Р 50.1.056 — 2005)
Менеджмент риска (risk management)	Полный процесс идентификации, контроля, устранения или уменьшения последствий опасных событий, которые могут оказать влияние на ресурсы информационно-телекоммуникационных технологий (ГОСТ Р ИСО/МЭК 13335-1 — 2006)

Термин	Содержание (ссылка)
Мониторинг безопасности информации	Постоянное наблюдение за процессом обеспечения безопасности информации в информационной системе с целью установить его соответствие требованиям безопасности информации (ГОСТ Р 50922 — 2006)
Недекларированные возможности (программного обеспечения)	Функциональные возможности программного обеспечения, не описанные в документации (ГОСТ Р 51275 — 2006)
Неотказуемость (non-repudiation)	Способность удостоверять имевшее место действие или событие так, чтобы эти события или действия не могли быть позже отвергнуты (ГОСТ Р ИСО/МЭК 13225-1 — 2006)
Несанкционированное блокирование доступа к информации (ресурсам информационной системы); отказ в обслуживании (denial of service)	Создание условий, препятствующих доступу к информации (ресурсам информационной системы) субъекту, имеющему право на него. Примечания: 1. Несанкционированное блокирование доступа осуществляется нарушителем безопасности информации, а санкционированное — администратором. 2. Создание условий, препятствующих доступу к информации (ресурсам информационной системы), может быть осуществлено по времени доступа, функциям по обработке информации (видам доступа) и (или) доступным информационным ресурсам (Р 50.1.056 — 2005)
Несанкционированное воздействие на информацию (ресурсы информационной системы); НСВ	Изменение, уничтожение или копирование информации (ресурсов информационной системы), осуществляемое с нарушением установленных прав и (или) правил. Примечания: 1. Несанкционированное воздействие может быть осуществлено преднамеренно или непреднамеренно. Преднамеренные несанкционированные воздействия являются специальными воздействиями. 2. Изменение может быть осуществлено в форме замены информации (ресурсов информационной системы), введения новой информации (новых ресурсов информационной системы),

Термин	Содержание (ссылка)
	а также уничтожения или повреждения информации (ресурсов информационной системы) (Р 50.1.056 — 2005)
Несанкционированное воздействие на информацию	Воздействие на защищаемую информацию с нарушением установленных прав и (или) правил доступа, приводящее к утечке, искажению, подделке, уничтожению, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации (ГОСТ Р 50922 — 2006)
Несанкционированный доступ к информации (ресурсам информационной системы); НСД	Доступ к информации (ресурсам информационной системы), осуществляемый с нарушением установленных прав и (или) правил доступа к информации (ресурсам информационной системы) с применением штатных средств информационной системы или средств, аналогичных им по своим функциональному предназначению и техническим характеристикам. Примечания: 1. Несанкционированный доступ может быть осуществлен преднамеренно или непреднамеренно. 2. Права и правила доступа к информации и ресурсам информационной системы устанавливают для процессов обработки информации, ее обслуживания, изменения программных, технических и информационных ресурсов, а также получения информации о них (Р 50.1.056 — 2005)
Норма эффективности защиты информации	Значение показателя эффективности защиты информации, установленное нормативными и правовыми документами (ГОСТ Р 50922 — 2006)
Обработка риска (risk treatment)	Процесс выбора и осуществления мер по модификации риска (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Объект защиты информации	Информация или носитель информации, или информационный процесс, которые необходимо защитить в соответствии с целью защиты информации (ГОСТ Р 50922 — 2006)

Термин	Содержание (ссылка)
Опасность	Потенциальный источник возникновения ущерба. Примечание. Термин «опасность» может быть конкретизирован в части определения природы опасности или вида ожидаемого ущерба (например, опасность электрического шока, опасность разрушения, травматическая опасность, техническая опасность, опасность пожара, опасность утонуть) (ГОСТ Р 51898 — 2002)
Статочный риск (residual risk)	Риск, остающийся после его обработки (ГОСТ Р ИСО/МЭК 13335-1 — 2006). Риск, остающийся после принятия защитных мер (ГОСТ Р 51898 — 2002)
Оценка информационного риска	Общий процесс анализа информационного риска и его оценивания (ГОСТ Р 50922 — 2006)
Оценка риска (risk assessment)	Процесс, объединяющий идентификацию риска, анализ риска и оценивание риска (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Оценка риска; анализ риска (risk assessment, risk analysis)	Выявление угроз безопасности информации, уязвимостей информационной системы, оценка вероятностей реализации угроз с использованием уязвимостей и оценка последствий реализации угроз для информационной системы, используемой для обработки этой информации (Р 50.1.056 — 2005)
Показатель эффективности защиты информации	Мера или характеристика для оценки эффективности защиты информации (ГОСТ Р 50922 — 2006)
Политика безопасности организации (organizational security policies)	Совокупность правил, процедур, практических приемов или руководящих принципов в области безопасности, которыми руководствуется организация в своей деятельности (ГОСТ Р ИСО/МЭК 15408 — 2008)

Термин	Содержание (ссылка)
Политика безопасности (информации в организации) (organisational security policy)	Совокупность документированных правил, процедур, практических приемов или руководящих принципов в области безопасности информации, которыми руководствуется организация в своей деятельности (ГОСТ Р 50922 — 2006)
Политика безопасности информационно-телекоммуникационных технологий (политика безопасности ИТТ) (ICT security policy)	Правила, директивы, сложившаяся практика, которые определяют, как в пределах организации и ее информационно-телекоммуникационных технологий управлять, защищать и распределять активы, в том числе критичную информацию (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Потенциал нападения (attack potential)	Предполагаемая возможность успеха в случае реализации нападения, выраженная в терминах квалификации, ресурсов и мотивации нарушителя (ГОСТ Р ИСО/МЭК 15408 — 2008)
Правила разграничения доступа (в информационной системе)	Правила, регламентирующие условия доступа субъектов доступа к объектам доступа в информационной системе (Р 50.1.056 — 2005)
Программная закладка	Преднамеренно внесенный в программное обеспечение функциональный объект, который при определенных условиях инициирует реализацию недекларированных возможностей программного обеспечения. Примечание. Программная закладка может быть реализована в виде вредоносной программы или программного кода (ГОСТ Р 51275 — 2006)
Программная закладка	Скрытно внесенный в программное обеспечение функциональный объект, который при определенных условиях способен обеспечить несанкционированное программное воздействие. Примечание. Программная закладка может быть реализована в виде вредоносной программы или программного кода (Р 50.1.056 — 2005)

Термин	Содержание (ссылка)
Программное воздействие	Несанкционированное воздействие на ресурсы автоматизированной информационной системы, осуществляемое с использованием вредоносных программ (ГОСТ Р 51275 — 2006)
Риск (risk)	Потенциальная опасность нанесения ущерба организации в результате реализации некоторой угрозы с использованием уязвимостей актива или группы активов. Примечание. Определяется как сочетание вероятности события и его последствий (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Санкционирование доступа; авторизация (authorization)	Предоставление субъекту прав на доступ, а также предоставление доступа в соответствии с установленными правами на доступ (Р 50.1.056 — 2005)
Сетевая атака	Компьютерная атака с использованием протоколов межсетевого взаимодействия (ГОСТ Р 51275 — 2006)
Система защиты информации	Совокупность органов и (или) исполнителей, используемой ими техники защиты информации, а также объектов защиты информации, организованная и функционирующая по правилам и нормам, установленным соответствующими документами в области защиты информации (ГОСТ Р 50922 — 2006)
Событие информационной безопасности (information security event)	Идентифицированное возникновение состояния системы, услуги или сети, указывающее на возможное нарушение политики информационной безопасности, отказ защитных мер, а также возникновение ранее неизвестной ситуации, которая может быть связана с безопасностью (ИСО/МЭК 18044 — 2004). Идентифицированное появление определенного состояния системы, сервиса или сети, указывающего на возможное нарушение политики информационной безопасности или отказ защитных мер, или возникновение неизвестной ранее ситуации, которая может иметь отношение к безопасности (ГОСТ Р ИСО/МЭК 18044 — 2007)

Термин	Содержание (ссылка)
Способ защиты информации	Порядок и правила применения определенных принципов и средств защиты информации (ГОСТ Р 50922 — 2006)
Средство защиты информации	Техническое, программное, программно-техническое средство, вещество и (или) материал, предназначенные или используемые для защиты информации (ГОСТ Р 50922 — 2006)
Техника защиты информации	Средства защиты информации, в том числе средства физической защиты информации, криптографические средства защиты информации, средства контроля эффективности защиты информации, средства и системы управления, предназначенные для обеспечения защиты информации (ГОСТ Р 50922 — 2006)
Требование по защите информации	Установленное правило или норма, которая должна быть выполнена при организации и осуществлении защиты информации, или допустимое значение показателя эффективности защиты информации (ГОСТ Р 50922 — 2006)
Угроза (threat)	Потенциальная причина инцидента, который может нанести ущерб системе или организации (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Угроза (безопасности информации) (threat)	Совокупность условий и факторов, создающих потенциальную или реально существующую опасность нарушения безопасности информации (ГОСТ Р 50922 — 2006)
Ущерб	Нанесение физического повреждения или другого вреда здоровью людей, или вреда имуществу, или окружающей среде (ГОСТ Р 51898 — 2002)
Уязвимость (vulnerability)	Слабость одного или нескольких активов, которая может быть использована одной или несколькими угрозами (ГОСТ Р ИСО/МЭК 13335-1 — 2006)

Термин	Содержание (ссылка)
Уязвимость (информационной системы), брешь (vulnerability, breach)	Свойство информационной системы, обусловливающее возможность реализации угроз безопасности обрабатываемой в ней информации. Примечания: 1. Условием реализации угрозы безопасности обрабатываемой в системе информации может быть недостаток или слабое место в информационной системе. 2. Если уязвимость соответствует угрозе, то существует риск (ГОСТ Р 50922 — 2006)
Целостность (integrity)	Свойство сохранения правильности и полноты активов (ГОСТ Р ИСО/МЭК 13335-1 — 2006)
Целостность информации (integrity)	Состояние информации, при котором отсутствует любое ее изменение либо изменение осуществляется только преднамеренно субъектами, имеющими на него право (Р 50.1.056 — 2005)
Цель безопасности (security objective)	Сформулированное намерение противостоять идентифицированным угрозам и (или) удовлетворять идентифицированной политике безопасности организации и предположениям (ГОСТ Р ИСО/МЭК 15408 — 2008)
Эффективность защиты информации	Степень соответствия результатов защиты информации цели защиты информации (ГОСТ Р 50922 — 2006)

Список основных используемых RFC

Номер RFC	Наименование
RFC 1631	Egevang K., Srisuresh P. The IP Network Address Translator (NAT). RFC 1631, May 1994
RFC 1661	Simpson W., Ed. The Point-to-Point Protocol (PPP), STD 51, RFC 1661, July 1994
RFC 1928	Leech M., Ganis M., Lee Y., Kuris R., Koblas D., Jones L. SOCKS Protocol Version 5. RFC 1928, March 1996
RFC 2341	Valencia A., Littlewood M., Kolar T. Cisco Layer Two Forwarding (Protocol) «L2F». RFC 2341, May 1998
RFC 2401	Kent S., Atkinson R. Security Architecture for the Internet Protocol. RFC 2401, November 1998
RFC 2402	Kent S., Atkinson R. IP Authentication Header. RFC 2402, November 1998
RFC 2406	Kent S., Atkinson R. IP Encapsulating Security Protocol (ESP). RFC 2406, November 1998
RFC 2407	Piper D. The Internet IP Security Domain of Interpretation for ISAKMP. RFC 2407, November 1998
RFC 2408	Maughan D., Schertler M., Schneider M., Turner J. Internet Security Association and Key Management Protocol (ISAKMP). RFC 2408, November 1998
RFC 2409	Harkins D., Carrel C. The Internet Key Exchange (IKE). RFC 2409, November 1998
RFC 2412	Orman H. The OAKLEY Key Determination Protocol. RFC 2412, November 1998
RFC 2459	Housley R., Ford W., Polk W., Solo D. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. RFC 2459, January 1999
RFC 2510	Adams C., Farrell S. Internet X.509 Public Key Infrastructure Certificate Management Protocols. RFC 2510, March 1999

Номер RFC	Наименование
RFC 2637	Hamzeh K., Pall G., Verthein W., Taarud J., Little W., G. Zorn. Point-to-Point Tunneling Protocol (PPTP). RFC 2637, July 1999
RFC 2661	Townsley W., Valencia A., Rubens A., Pall G., Zorn G., Palter B. Layer Two Tunneling Protocol 'L2TP'. RFC 2661, August 1999.
RFC 2764	Gleeson B., Lin A., Heinanen J., Armitage G., Malis A. A Framework for IP Based Virtual Private Networks. RFC 2764, February 2000
RFC 2766	Tsirtsis P. Network Address Translation — Protocol Translation (NAT-PT), RFC 2766, February 2000
RFC 5830	Dolmatov V., Ed. GOST 28147—89: Encryption, Decryption and Message Authentication Code (MAC) Algorithms. RFC 5830, March 2010
RFC 5831	Dolmatov V., Ed. GOST R 34.11—94: Hash Function Algorithm RFC 5831, March 2010
RFC 5832	Dolmatov V., Ed. GOST R 34.10—2001: Digital Signature Algorithm. RFC 5832, March 2010

СПИСОК ЛИТЕРАТУРЫ

1. *Белов Е. Б.* Основы информационной безопасности / Е. Б. Белов, В. П. Лось, Р. В. Мещеряков, А. А. Шелупанов. — М.: Горячая линия — Телеком, 2006.

2. *Бил Дж.* Snort 2.1. Обнаружение вторжений. — 2-е изд. / Бил Дж. и др.; пер. с англ. — М.: Бином-Пресс, 2006.

3. *Блэк У.* Интернет: протоколы безопасности: учебный курс / У. Блэк. — Спб.: Питер, 2001.

4. *Дюк В.* Data Mining : учебный курс / В. Дюк, А. Самойленко. — СПб. : Питер, 2001.

5. *Зима В. М.* Безопасность глобальных сетевых технологий — 2-е изд. / В. М. Зима, А. А. Молдовян, Н. А. Молдовян. — СПб. : БХВ-Петербург, 2003.

6. *Ибе О.* Сети и удаленный доступ. Протоколы, проблемы, решения : пер. с англ. / О. Ибе. — М. : ДМК Пресс, 2002.

7. Инструменты, тактика и мотивы хакеров. Знай своего врага : пер. с англ. / коллектив авторов. — М. : ДМК Пресс, 2003.

8. *Лукацкий А. В.* Обнаружение атак / А. В. Лукацкий. — СПб. : БХВ-Петербург, 2001.

9. Новый словарь хакера: пер. с англ. / под ред. Р. С. Рэймонда. — М.: ЦентрКом, 1996.

10. *Норткатт С.* Анализ типовых нарушений безопасности в сетях : пер. с англ. / С. Норткатт, М. Купер, М. Фирноу, К. Фредерик. — М.: Вильямс, 2001.

11. *Норткатт С.* Защита сетевого периметра : пер. с англ. / С. Норткатт и др. — К. : ТИД «ДС», 2004.

12. *Норткатт С.* Обнаружение вторжений в сеть. Настольная книга специалиста по системному анализу / С. Норткатт, Дж. Новак, Д. Маклахлен. — М. : ЛОРИ, 2001.

13. *Осовский С.* Нейронные сети для обработки информации / С. Осовский ; пер. с польск. И. Д. Рудинского. — М. : Финансы и статистика, 2002.

14. *Петренко С. А.* Управление информационными рисками. Экономически оправданная безопасность / С. А. Петренко. — М. : ДМК Пресс, 2004.

15. *Польман Н.* Архитектура брандмауэров для сетей предприятия : пер. с англ. / Н. Польман, Т. Кразерс. — М. : Вильямс, 2003.

16. *Поляков А.* Безопасность Oracle глазами аудитора: Нападение и защита. — М.: ДМК Пресс, 2009.

17. *Ростовцев А. Г.* Теоретическая криптография / А. Г. Ростовцев, Е. Б. Маховенко. — СПб. : Профессионал, 2005.

18. *Рутковская Д.* Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский; пер. с польск. И. Д. Рудинского. — М.: Горячая линия — Телеком, 2004.

19. *Соломон Д.* Внутреннее устройство Microsoft Windows 2000/ Мастер класс. /Д. Соломон, М. Руссинович; пер. с англ. — Спб.: Питер; М.: Русская Редакция, 2001.

20. Средства вычислительной техники. Межсетевые экраны. Защита — несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации : руководящий документ, с М., 1998.

21. *Столингс В.* Основы защиты сетей. Приложения и стандарты : пер. англ. / В. Столингс. — М. : Вильямс, 2002.

22. *Форд Д. Л.* Персональная защита от хакеров. Руководство для начинающих : пер. с англ. / Д. Л. Форд. — М. : КУДИЦ-ОБРАЗ, 2002.

23. *Хабракен Д.* Маршрутизаторы Cisco. Практическое применение : пер. с англ. / Д. Хабракен. — М. : ДМК Пресс, 2001.

ОГЛАВЛЕНИЕ