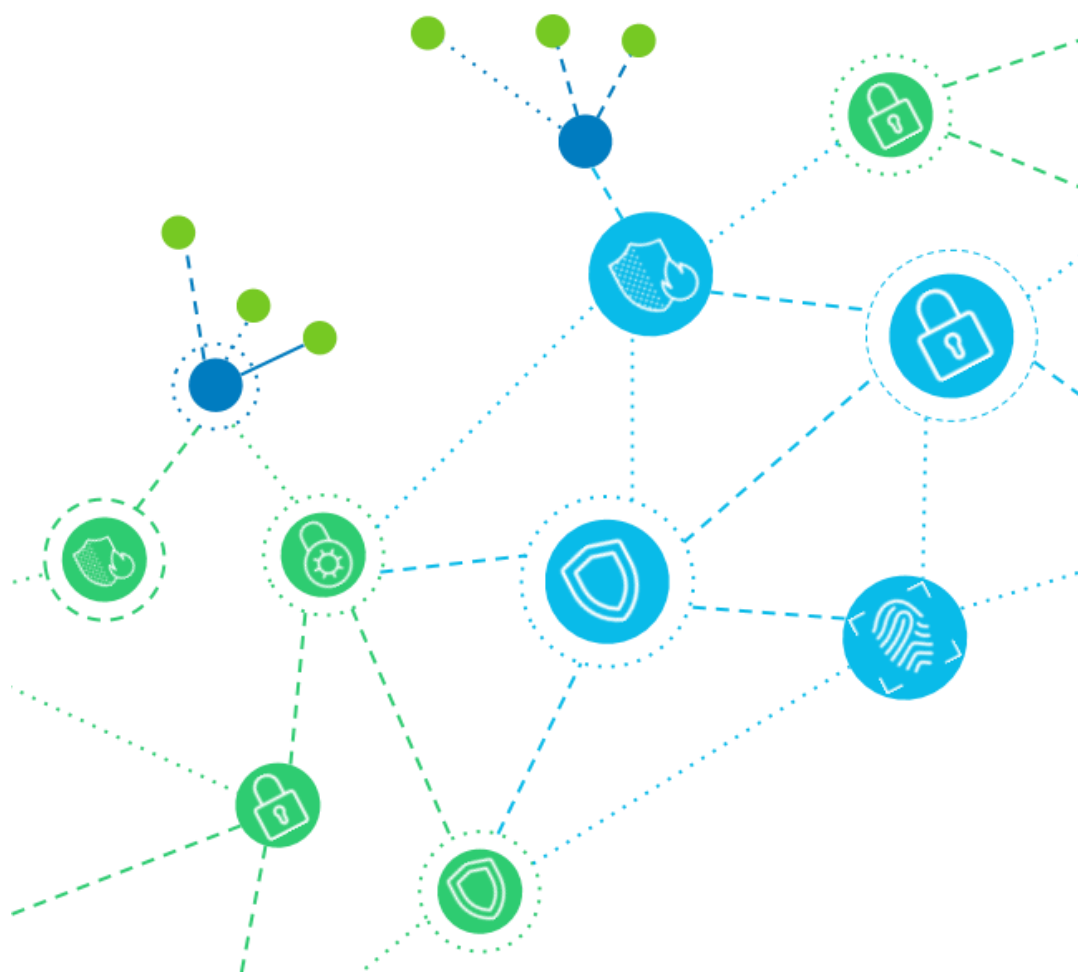


Для тех, кто не знает с чего начать...

Практическая безопасность сетей

Ольков Евгений, 2017
blog.netskills.ru



От автора	6
О чем эта книга?	6
Для кого эта книга?	7
Применение	8
Введение	9
Миф безопасности	9
1. Доступ к оборудованию	11
1.1 Физический доступ	11
1.1.1 Вход в систему	12
1.1.2 Задаем пароль на enable	14
1.1.3 Создание пользователей	18
1.1.4 Подключение по консоли	20
1.1.5 Сброс пароля	21
1.2 Удаленный доступ	22
1.2.1 VTU	23
1.2.2 Telnet	25
1.2.3 SSH	26
1.2.4 HTTP/S	29
1.2.5 Ограничение доступа	31
1.2.6 Списки доступа к оборудованию	32
1.2.7 Защита от Brute Force	33
1.2.8 Идентификация устройства	34
1.3 AAA	36
1.3.1 RADIUS	37
1.3.2 TACACS+	38
1.3.3 RADIUS vs TACACS+	38
1.3.4 Настройка AAA-сервера	39
1.3.5 Настройка Authentication	42
1.3.6 Настройка Authorization	44
1.3.7 Настройка Accounting	46
1.3.8 Ограничения AAA	46
1.4 Парольная политика	47
1.5 Чек-лист №1	51
1.6 Пример конфигурации	52
2. Лучшие практики	54
2.1 Logs	54
2.1.1 Методы сбора логов	55
2.1.2 Уровни логирования	55

2.1.3 Console Logging	56
2.1.4 Buffered Logging	56
2.1.5 Terminal Logging	57
2.1.6 Syslog-сервер	57
2.1.7 SNMP Traps	59
2.1.8 Безопасность логов	59
2.1.9 Время	60
2.2 Лишние сервисы	61
2.3 Резервная память	62
2.4 Защищенные протоколы	62
2.4.1 SCP	62
2.4.2 SNMP	63
2.5 Резервные копии	64
2.6 Логирование команд	66
2.7 Обновления	66
2.8 Чек-лист №2	67
2.9 Пример конфигурации	68
3. Защищаем локальную сеть	70
3.1 Уровень доступа	71
3.1.1 Сегментация	71
3.1.2 Неиспользуемые порты	73
3.1.3 VLAN1	73
3.1.4 DTP	75
3.1.5 Trunk - порты	76
3.1.6 Имена портов и VLAN	78
3.1.7 Штормы	79
3.1.8 PortFast	80
3.1.9 BPDU Guard	81
3.1.10 Port Security	83
3.1.11 DHCP snooping	84
3.1.12 IP Source Guard	86
3.1.13 Dynamic ARP Inspection	87
3.1.14 IEEE 802.1X	87
3.2 Уровень распределения и ядра	88
3.2.1 STP priority	88
3.2.2 Root Guard	89
3.2.3 RSTP	89
3.2.4 Общие рекомендации	90
3.3 Чеклист №3	90
3.4 Пример конфигурации	91
4. Управление безопасностью	95

4.1 Иерархия средств защиты	96
4.2 Политика безопасности	97
4.2.1 Верхний уровень	98
4.2.2 Средний уровень	98
4.2.3 Нижний уровень	99
4.2.4 Пример политики безопасности	99
4.4 Чеклист №4	100
5. Управление доступом	101
5.1 С чего начать?	101
5.2 Что защищать?	102
5.3 От чего защищать?	102
5.4 Как защищать?	103
5.5 Чем защищать?	104
5.6 Списки доступа	105
5.6.1 Матрица прав доступа	106
5.6.2 Методы назначения прав. RBAC	108
5.6.3 VLAN и RBAC	109
5.6.4 Порты	109
5.6.5 Входящий или исходящий	111
5.6.6 Два главных подхода	112
5.7 Лучшие практики	113
5.7.1 Именованные списки доступа	114
5.7.2 Логирование правил	114
5.7.3 Топ-правила	115
5.7.4 Неиспользуемые правила	116
5.7.8 Паразитный трафик	116
5.8 Обратные списки доступа	117
5.9 Чеклист №5	117
5.10 Пример конфигурации	118
6. Защита периметра	120
6.1 Stateful Packet Inspection	121
6.2 Zone-Based Firewall	123
6.2.1 Подготовка (NetFlow)	124
6.2.2 Zone	126
6.2.3 Zone-pair	126
6.2.4 Class-map	127
6.2.5 Policy-map	129
6.2.6 Service-policy	129
6.2.7 Интерфейсы	130
6.2.8 DoS	130
6.3 DMZ	131

6.3.1 Зачем?	131
6.3.2 Лишние сервисы	132
6.3.3 Private VLAN	133
6.4 Чеклист №6	136
6.5 Пример конфигурации	136
Заключение	141

*Посвящается моей жене Ксении и маленькому сыну Степану, которые всегда
меня поддерживали и были моей самой главной мотивацией.*

От автора

Данная книга является результатом нескольких лет работы в области системной интеграции, а также основана на анализе и переработке огромного количества профессиональной литературы. Руководство носит исключительно информативный характер. Приведенные рекомендации всего лишь личное мнение автора и не являются абсолютной истиной!

Я буду придерживаться свободного стиля изложения и возможно у вас иногда будет возникать ощущение, что вы читаете художественную, а не техническую литературу. Надеюсь, что это позволит гораздо легче воспринимать весь материал.

Автор имеет техническое образование и совершенно однозначно не может назвать себя писателем. Книга не проходила никакой модерации, поэтому с большой долей вероятности вы встретите как орфографические, так и пунктуационные ошибки. Заранее прошу прощения (буду благодарен, если вы сообщите мне об ошибках в тексте)!

Кроме того, хотел бы обратиться к читателям с просьбой не распространять эту книгу в сети Интернет. Автор потратил очень много сил на ее написание. Хоть и небольшой доход от продаж книги позволяет поддерживать и развивать бесплатный проект NetSkills, где и в дальнейшем будут публиковаться бесплатные видео курсы.

О чем эта книга?

Если взять мою предыдущую книгу “Архитектура корпоративных сетей”, то можно узнать практически о всех средствах защиты сети. Однако следует понимать, что в той книге описывается идеальный случай, когда вы не ограничены бюджетом и можете покупать все, что вам нужно. Также стоит помнить, что даже самый дорогой межсетевой экран не сможет защитить вашу сеть, если вы не будете его настраивать должным образом. Информационная безопасность это не результат, это непрерывный процесс. Нельзя установить средства защиты и решить, что теперь ваша сеть безопасна. Вам придется постоянно дорабатывать свою систему защиты.

Мы живем в реальном мире и далеко не всегда располагаем нужными средствами в ИТ/ИБ бюджете для закупки тех или иных средств. Что же делать тем, у кого отсутствует какое-либо оборудование для защиты сети? Неужели придется поставить крест на безопасности? Многие так и делают, говоря: “О какой безопасности может идти речь, если компания не хочет или не может выделять средства”. С этой мыслью многие почему-то совершенно забывают о встроенных средствах защиты в обычном сетевом оборудовании, не соблюдают простейшие правила безопасности и оставляют огромные “дыры” в своей сети. Не делайте так. Если вы беретесь за какую-то работу, то делайте ее максимально хорошо, используя все доступные возможности.

По работе мне часто приходилось участвовать в очень крупных проектах, где ставилась задача по защите сети в 5-10 тысяч пользователей. Закупалось огромное количество дорогого оборудования - межсетевые экраны, системы предотвращения

вторжений, прокси-сервера и т.д. Бюджет проектов исчислялся десятками миллионов рублей. Представьте мое удивление, когда после внедрения таких “дорогущих” проектов обнаруживалось, что на обычном сетевом оборудовании использовались пароли вроде “admin” или “1234” (и эти пароли не менялись годами, даже после смены системных администраторов). Для подключения к коммутаторам использовался незащищенный протокол “Telnet”. В офисах стояли хабы, которые принесли сами пользователи, “потому что им так удобнее”. В корпоративную сеть подключались личные ноутбуки сотрудников. В ИТ инфраструктуре была полная анархия.

Таким образом, несмотря на потраченные миллионы, такую сеть мог бы “положить” даже школьник в течении 5 минут. А все из-за безалаберного отношения к обычным коммутаторам и маршрутизаторам, которые обычно никто даже не рассматривает в качестве средств защиты.

В данной книге мы узнаем каким образом можно “закрутить гайки безопасности” на обычных коммутаторах и маршрутизаторах. Попробуем описать лучшие практики по настройке оборудования и процессы, необходимые для обеспечения информационной безопасности.

Для кого эта книга?

Современный работодатель считает (или хочет так считать) что ИТ-специалист и специалист по ИБ, это одно и то же. Безусловно это не так. Это совершенно разные профессии. Но в силу различных обстоятельств (не зависящих от нас), очень часто приходится совмещать эти две профессии. Возможно кого-то возмутит данный факт, но я убежден, что любой уважающий себя ИТ-специалист должен обладать хотя бы элементарными знаниями в области сетевой безопасности и придерживаться некоторых принципов при построении или администрировании сети.

Если перед вами вдруг встала задача обезопасить свою сеть, но тема сетевой безопасности для вас темный лес, то стоит начать именно с этой книги.

Безусловно, это руководство не сделает из вас эксперта в области ИБ. Но мы и не ставим такой цели. Эта книга для тех, кто хочет в кратчайшие сроки улучшить защиту своей сети. Выжмите максимум из имеющегося оборудования, а уж затем можно думать о таких вещах как DLP, SIEM, Proxy, IPS и т.д. Строительство дома всегда начинается с фундамента. В нашем случае, фундамент безопасности - грамотно настроенное сетевое оборудование и выстроенный процесс сопровождения сети.

Эта книга будет вам интересна, если вы:

1. хотите узнать об основных опасностях для вашей сети;
2. хотите узнать как от них защищаться;
3. вам нужны практические советы по защите сети;
4. хотите стандартизировать настройки безопасности для сетевого оборудования;

5. вам интересны лучшие практики в области сетевой безопасности;
6. просто интересуетесь сетевой безопасностью.

Также это руководство отлично подойдет для обучения сотрудников внутри организации - сетевых инженеров и системных администраторов. Данная книга особенно рекомендуется для завершивших "[Курс молодого бойца](#)", как логическое продолжение основ сетевых технологий.

Применение

Данное руководство описывает основные аспекты построения защищенной сети на основе коммутаторов и маршрутизаторов, без использования специализированных средств (межсетевых экранов, IPS, DLP и т.д.).

В книге будут даны не только рекомендации, но и конкретные примеры конфигурации оборудования. Все примеры будут касаться исключительно оборудования компании Cisco Systems.

Это не значит, что если вы используете коммутаторы другой фирмы, то данная книга вам не подойдет. Все описанные настройки присутствуют в подавляющем большинстве сетевого оборудования. Принцип остается тем же, отличаться будет только синтаксис команд.

Следует отметить, что описанные в книге методы не гарантируют абсолютной безопасности вашей сети. Руководство дает некий "шаблон", придерживаясь которого вы сможете существенно обезопасить сетевую инфраструктуру.

Также предполагается, что читатель обладает необходимым уровнем знаний и способен отличить коммутатор от маршрутизатора.

После прочтения данного руководства настоятельно рекомендую к ознакомлению книгу "Архитектура корпоративных сетей". Книга познакомит вас с принципами построения более комплексной и надежной защиты на основе специализированных продуктов.

Важное замечание! Абсолютно все описанные технологии и настройки, можно попробовать в **GNS3**, **UNetLab (EVE NG)** или **Cisco Packet Tracer**. Если у вас нет опыта работы с данными программами, то вы можете воспользоваться бесплатными видео уроками, которые вы найдете на сайте **blog.netskills.ru**.

Введение

Информационная безопасность (ИБ). Это очень емкое понятие, которое трактуется по-разному. Но почти всегда эту фразу ассоциируют с чем-то сложным, непонятным и даже раздражающим. Некоторые пользователи вообще ненавидят эту самую безопасность, особенно когда им неожиданно закрыли доступ к любимому сайту или ограничили права на файловом хранилище. Большинство воспринимают ИБ как нечто мифическое, пока сами не столкнутся с неприятностями. А учитывая последние тенденции компьютеризации всего и вся, возможностей у злоумышленников становится все больше, в то время как мы становимся все более и более уязвимыми.

Многие часто используют понятие “сетевая безопасность” как синоним “информационной безопасности”. Это в корне не верно. Под сетевой безопасностью мы будем подразумевать защиту нашей ИТ - инфраструктуры от злоумышленников (как внешних так и внутренних), а также защиту от случайных ошибок персонала. Да, опасность исходит не только от хакеров. Наш собственный пользователь представляет не меньшую опасность, сам того не подозревая.

Мы начнем с базовых вещей, которые можно сделать “здесь и сейчас”, без серьезного переделывания сети. Уделим внимание типичным ошибкам администраторов сети. Узнаем об основных угрозах для сети и каким образом от них защититься. Рассмотрим лучшие практики по настройке оборудования.

К концу книги мы рассмотрим более сложные вещи. Научимся производить простейший аудит сети. Узнаем каким образом разграничивать доступ к корпоративным ресурсам и попробуем разработать свою первую матрицу доступа.

В итоге мы получим некий “чек-лист”, на который следует ориентироваться при построении безопасной сети.

Миф безопасности

Прежде чем продолжись повествование, я должен сделать небольшое лирическое отступление.

Если вас захотят взломать, вы не сможете этому противостоять. Чтобы вы не делали.

Возможно данный тезис не очень мотивирует к дальнейшему чтению, но нужно правильно понимать смысл этого утверждения. Ни одна система защиты не даст 100% гарантию. Пока вы дочитали до этой строки уже появилось несколько новых (ранее неизвестных) уязвимостей, которые уже кто-то использует с весьма корыстными целями. Обеспечение информационной безопасности похоже на гонку вооружений, вот только хакер всегда на шаг впереди. Всегда будут первые жертвы, после которых появляются описания данных брешей, выходят всевозможные патчи и обновления для

средств защиты. Периодически взламываются такие серьезные структуры как ФБР, Пентагон, ФСБ, Kaspersky, где бюджеты ИБ исчисляются миллиардами, а в службе безопасности работают лучшие из лучших. Но даже они не могут защититься от таргетированных (целенаправленных) атак, когда высококлассный хакер кропотливо пытается подобрать ключик к самой современной системе защиты. Как бы дико не звучало, но даже полностью изолированные сети (отключенные от Интернета) подвержены успешным атакам. Современные вирусы способны внедряться в закрытые сети через флеш-носители сотрудников, собирать нужную информацию и терпеливо ждать, когда они смогут выбраться через те же флешки.

Но зачем тогда защищаться, если все так плохо? Но все плохо только на первый взгляд. Процент таких таргетированных атак ничтожно мал, по сравнению с атаками “на дурака”. Подавляющее большинство вторжений в сеть происходит с применением простейших и давно известных приемов. Для воспроизведения таких атак не требуется быть хакером - экспертом. Достаточно скачать специализированные дистрибутивы (например Kali Linux) и воспользоваться уже готовыми программами для взлома. Не нужно глубокое понимание работы стека TCP/IP, не нужно уметь программировать, не нужно разбираться в современных средствах защиты. Просто скачать дистрибутив, нажать пару кнопок и готово. Можно даже скачать готовый вирус/троян и закинуть его по почте. Наблюдается интересная статистика, в праздничные дни возрастает кол-во атак. Как вы думаете, почему? Из-за студентов и школьников на каникулах, которые пробуют свои силы в хакинге с помощью популярных и весьма доступных утилит.

Кроме того, как было сказано выше, угрозу представляют не только внешние “враги”, но и внутренние пользователи. Неосторожные действия могут привести к весьма длительному простоя сети, либо открыть окно для внешнего злоумышленника. Поэтому не стоит пенять на отсутствие средств защиты, забывая о грамотной настройке основного сетевого оборудования. Даже на уровне коммутаторов и маршрутизаторов вы сможете отсеять подавляющее большинство опасностей для вашей сети.

1. Доступ к оборудованию

Ограничить доступ к сетевому оборудованию, как физически, так и удаленно. Это первое с чего вы должны начать. Просто поразительно сколько неприятностей происходит в корпоративных сетях только из-за неограниченного доступа к оборудованию.

1.1 Физический доступ

Абсолютно в каждой компании найдется сотрудник, который считает себя намного умнее системного администратора. И когда ему кажется, что в сети возникли какие-то проблемы (или же ему захотелось поменять местами пару проводов) он уверенно идет к коммутатору и делает все, что душе угодно (вплоть до перезагрузки оборудования). В результате в сети возникают “петли”, либо просто путаница с подключениями. Данный инцидент можно смело причислять к “вторжениям”. Это прямая опасность для вашей сети.

В идеале все сетевое оборудование должно находиться в серверной комнате, которая закрывается на ключ и доступ туда имеет лишь ограниченный круг лиц. Но мы живем не в идеальном мире и очень часто коммутаторы оказываются либо в коридоре помещения, либо вовсе в кабинете под столом. Отличным решением в данном случае будет установка специализированных телекоммуникационных шкафов. Наиболее удобны небольшие настенные шкафы имеющие замок (см. рис. X).



Рис. X. Настенный 19 - и дюймовый шкаф.

Естественно, установленный замок не поможет от целенаправленной попытки добраться до сетевого оборудования. Но мы и не ставим такой цели. Данный шкаф это скорее “защита от дурака”. Вряд ли офисный сотрудник начнет ломать замок желая переключить пару проводов. Именно это нам и нужно. В случае невозможности

установки подобного шкафа, постарайтесь убрать сетевое оборудование в максимально недоступное место: под потолок, на высокий шкаф и т.д.

Если же ваше сетевое оборудование находится в неконтролируемой зоне (общие коридоры, лестничные пролеты), то телекоммуникационный шкаф с надежным замком является обязательным условием. Данная мера позволит существенно снизить риск физического доступа злоумышленника к вашей сети. Иными словами - хакер не сможет подключиться патч-кордом к вашему коммутатору. Плюс это может спасти вас от вандалов, которые также представляют угрозу для информационной безопасности.

Тоже самое можно сказать про витую пару. Все провода должны проходить в труднодоступных местах, чтобы снизить вероятность злонамеренного внедрения.

Ограничьте физический доступ к вашему сетевому оборудованию, а также кабельной системе. Это снизит риски как случайных, так и целенаправленных вторжений.

1.1.1 Вход в систему

Для входа в саму операционную систему Cisco IOS есть два способа: через консольный порт и удаленно (протоколы telnet, ssh, https). Прежде чем рассмотреть вопрос удаленного доступа необходимо описать некоторые моменты, касающиеся входа через консоль и в Cisco IOS в целом.

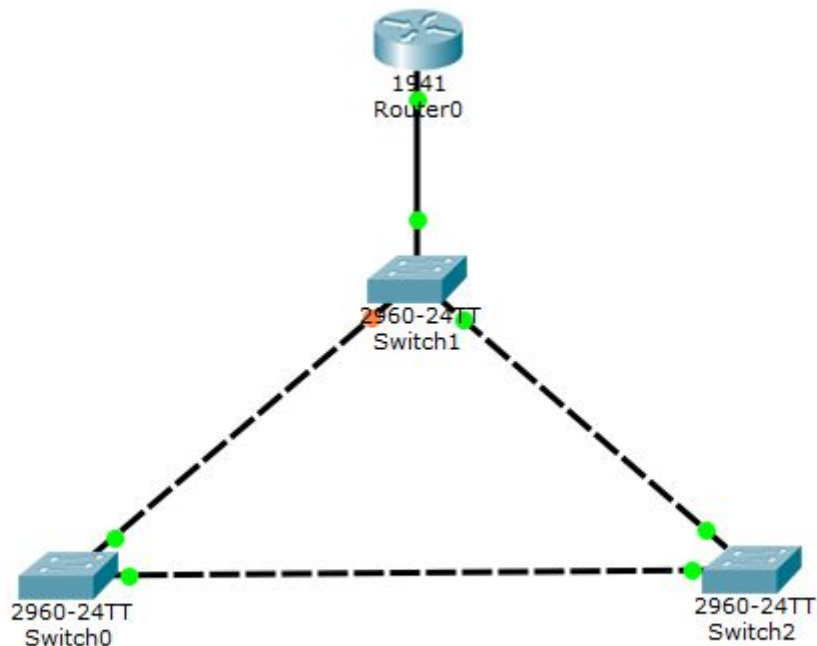
Возможно шкаф с замком не остановил злоумышленника и он получил прямой физический доступ к коммутатору. Первое что он попытается сделать - подключиться по консоли. Именно поэтому очень важно принять меры по защите консольного подключения.

Наверняка большинство читателей уже немного знакомы с Cisco IOS, но я все же расскажу несколько базовых вещей. Из соображений безопасности в Cisco IOS существует два режима доступа к командной строке:

1. Пользовательский режим (User EXEC Mode). Если вы зашли на коммутатор или маршрутизатор и видите приглашение вроде "**Switch>**", т.е. строка оканчивающаяся на знак ">", то вы именно в пользовательском режиме. Из данного режима невозможно производить настройку оборудования или просмотреть текущую конфигурацию.
2. Привилегированный режим (Privileged EXEC Mode). В данном режиме приглашение выглядит следующим образом - "**Switch#**", т.е. заканчивается на знак "#". Для входа в данный режим необходимо ввести команду **enable**. Из данного режима можно производить настройку оборудования и просматривать конфигурацию.

Многие почему-то пренебрегают защитой пользовательского режима и оставляют вход по консоли совершенно открытым, ограничиваясь лишь паролем на привилегированный режим (т.е. пароль на **enable**). Казалось бы, из User mode злоумышленник не сможет поправить или просмотреть конфигурацию. Но это безопасно только на первый взгляд. Получив доступ к оборудованию из

пользовательского режима возможно собрать большое количество информации о сети в целом. Приведем простой пример. Пусть наша сеть выглядит следующим образом:



Предположим что злоумышленник подключился по консоли к коммутатору Switch1. Вход в привилегированный режим закрыт паролем. Для просмотра доступных команд ему достаточно набрать знак “?”:

```
Switch>?  
Exec commands:  
connect      Open a terminal connection  
disable      Turn off privileged commands  
disconnect    Disconnect an existing network connection  
enable        Turn on privileged commands  
exit          Exit from the EXEC  
logout        Exit from the EXEC  
ping          Send echo messages  
resume        Resume an active network connection  
show          Show running system information  
telnet        Open a telnet connection  
terminal      Set terminal line parameters  
traceroute    Trace route to destination
```

Как видим довольно большой список. По умолчанию на всех устройствах Cisco включен протокол CDP. Это проприетарный протокол компании Cisco позволяющий обнаруживать подключенное напрямую сетевое оборудование (опять же компании Cisco). Какую же информацию можно получить набрав команду **show cdp neighbors detail**?

```
Switch>show cdp neighbors detail
```

```
Device ID: Switch  
Entry address(es):  
IP address : 192.168.1.3  
Platform: cisco 2960, Capabilities: Switch  
Interface: FastEthernet0/1, Port ID (outgoing port): FastEthernet0/1
```

Holdtime: 136

Version :

Cisco IOS Software, C2960 Software (C2960-LANBASE-M), Version 12.2(25)FX, RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2005 by Cisco Systems, Inc.
Compiled Wed 12-Oct-05 22:05 by pt_team

advertisement version: 2

Duplex: full

Device ID: Switch

Entry address(es):

IP address : 192.168.1.4

Platform: cisco 2960, Capabilities: Switch

Interface: FastEthernet0/2, Port ID (outgoing port): FastEthernet0/1

Holdtime: 161

Version :

Cisco IOS Software, C2960 Software (C2960-LANBASE-M), Version 12.2(25)FX, RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2005 by Cisco Systems, Inc.
Compiled Wed 12-Oct-05 22:05 by pt_team

advertisement version: 2

Duplex: full

Device ID: Router

Entry address(es):

IP address : 192.168.1.254

Platform: cisco C1900, Capabilities: Router

Interface: FastEthernet0/3, Port ID (outgoing port): GigabitEthernet0/0

Holdtime: 139

Version :

Cisco IOS Software, C1900 Software (C1900-UNIVERSALK9-M), Version 15.1(4)M4, RELEASE SOFTWARE (fc2)
Technical Support: <http://www.cisco.com/techsupport>
Copyright (c) 1986-2012 by Cisco Systems, Inc.
Compiled Thurs 5-Jan-12 15:41 by pt_team

advertisement version: 2

Duplex: full

Теперь злоумышленник знает практически все о нашей сети: ip адреса, модели устройств и даже версии их прошивки. Сведения о прошивке позволяют использовать известные уязвимости. Ситуация усугубляется тем, что далеко не все администраторы следят за выходом новых прошивок, в которых закрываются эти известные уязвимости. Кроме того, ему доступны такие команды как **ping**, **traceroute**, **telnet**, **show vlan**, **show arp** и т.д. В итоге он получает идеальную площадку для сбора информации и дальнейшей атаки, пытаюсь проникнуть на другие устройства (telnet).

Закрывайте доступ к консольному порту даже для пользовательского режима.

Ниже мы рассмотрим как сделать это правильно.

1.1.2 Задаем пароль на enable

Установка пароля на привилегированный режим, это первое с чего начинается настройка оборудования Cisco. Возможно многие читатели подумают, что я пишу об очевидных вещах. Однако, даже в этом простом шаге есть свои нюансы.

Большинство руководств в сети Интернет описывает процесс установки пароля с помощью команд:

```
Switch#configure terminal          /вход в режим глобальной конфигурации
Switch(config)#enable password cisco /установка пароля
```

Казалось бы, что на этом можно и закончить. Привилегированный режим защищен паролем. Отчасти да. Однако стоит знать, что данный пароль хранится в конфигурации устройства как обычный текст и если использовать команду **show run**, то можно увидеть следующее:

```
Switch#show running-config | include enable password
enable password cisco          /пароль хранится в открытом виде
```

Чем же это опасно? Возможно вы, как добросовестный администратор, храните резервные копии конфигураций, что вполне логично. Как правило такие резервные копии лежат на FTP либо TFTP сервере. Возможна ситуация, когда злоумышленник получил доступ к этому серверу. Просмотрев конфигурации устройств и найдя пароли, он получит доступ ко всей вашей сети. Другой вариант, когда вы открыли сессию удаленного доступа (telnet, ssh) к одному из коммутаторов, но вас вдруг позвали в другую комнату. Вы оставили окошко с открытой сессией буквально на 2 минуты. Возможно этого времени недостаточно, чтобы внести серьезные изменения в конфигурацию, но этого точно хватит, чтобы ваш коллега или случайный “прохожий” из другого отдела успел подглядеть пароль. Более того, подсмотреть пароль могут просто находясь у вас за спиной, когда вы просматриваете конфигурацию.

Очевидно, что пароль нужно хранить в зашифрованном виде. В этом случае в сети Интернет советуют использовать следующую команду:

```
Switch(config)#service password-encryption
```

Данная команда призвана шифровать все имеющиеся пароли в конфигурации устройства. Вот так выглядит пароль после применения команды:

```
Switch#show running-config | include enable password
enable password 7 0822455D0A16          /зашифрованный пароль
```

Как видим пароль уже выглядит совсем иначе. Это так называемые Type 7 пароли. Но и этого недостаточно. Дело в том, что команда **service password-encryption** использует довольно слабый и широко известный метод шифрования. Данные пароли очень просто поддаются дешифрации. В интернете огромное количество утилит, которые позволяют это сделать. Яркий пример - утилита Cain&Abel. Также доступны онлайн [ресурсы](#) (это всего лишь пример и вы можете найти другие сайты используя в google поисковый запрос “cisco type 7 password decrypt”). Давайте попробуем восстановить пароль из нашей зашифрованной последовательности 0822455D0A16.

Cisco Type 7 Reverser

Paste any Cisco IOS "type 7" password string into the form below to retrieve the plaintext value. Type 7 passwords appears as follows in an IOS configuration file. Copy and paste only the portion **bolded** in the example.

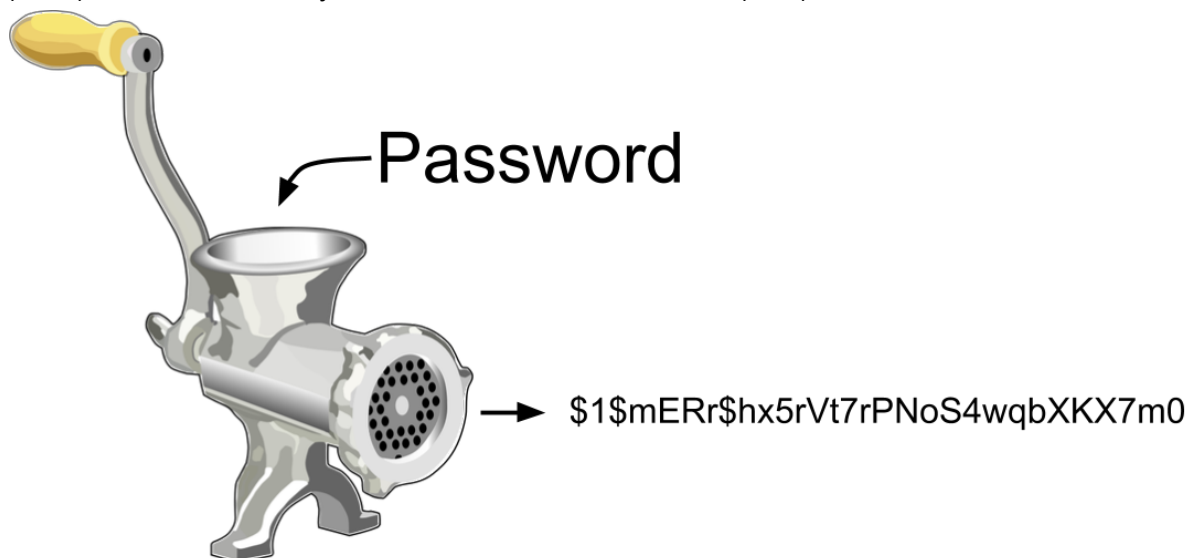
```
[...] password 7 046E1803362E595C260E0B240619050A2D
```

Type7 hash

Reversed

Пароль успешно восстановлен. Таким образом, если злоумышленник все же смог получить доступ к конфигурации устройства (или возможно он смог запомнить зашифрованный пароль одним лишь взглядом), то он с легкостью расшифрует реальный пароль. Шифрование здесь не помогло.

Для решения этой проблемы существует **enable secret**. Именно эту команду необходимо использовать при задании пароля. Функция secret использует совершенно другой принцип. Вместо шифрования, пароль проходит через определенный алгоритм (MD5) и на выходе получается так называемый hash (хэш).

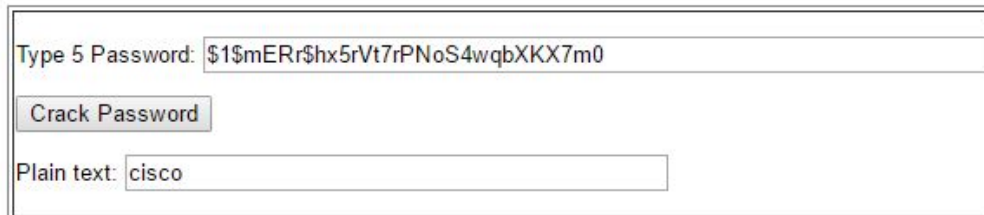


Это операция в одну сторону. Ее можно сравнить с мясорубкой. Если вы перекрутите кусок мяса в фарш, то обратно кусок мяса вы уже не получите никаким образом. В конфигурации отображается именно hash, который невозможно расшифровать. Для примера давайте зададим пароль и посмотрим, что получится на выходе:

```
Switch(config)#enable secret cisco
Switch#show running-config | include enable
enable secret 5 $1$mERr$hX5rVt7rPNoS4wqbXKX7m0 /hash исходного пароля "cisco"
```

В конфигурации мы видим hash нашего исходного пароля. Цифра “5” после **enable secret** означает, что используется алгоритм хэширования (в случае с **enable password** отображается цифра 7). Как было сказано выше, данный пароль невозможно декодировать. Однако и это не дает нам 100% защиты. Данный пароль можно брутфорсить (от англ. brute force), т.е. подбирать. В интернете существуют таблицы, где для самых распространенных паролей приведены их хэши. Есть даже специализированные сервисы, [например](#) (можете найти другой сервис в google, используя запрос “Cisco IOS Enable Secret Type 5 Password Cracker”).

Take the type 5 password, such as the text above in red, and paste it into the box below and click "Crack Password".



Have you got a type 7 password you want to break? Try our [Cisco type 7 password cracker](#) instead.

What's the moral of the story?

Don't use stupidly simple passwords.

Если поместить наш хэш (\$1\$mERr\$hX5rVt7rPNoS4wqbXKX7m0) в эту форму, то в поле “Plain text” мы получим исходный пароль менее чем за секунду. Т.е. данный сервис просто перебирает значения в таблице, где хранятся хэши часто используемых паролей. Если же в существующей таблице нет приведенного хэша, можно попробовать применить “атаку по словарю”. В этом случае подбирается пароль, затем генерируется его хэш и сравнивается с исходным. В этом может помочь утилита Cain&Abel. Преимущества данного брутфорса перед обычным (когда пароль подбирается непосредственно на устройстве) в том, что вы не увидите данный процесс в логах своего устройства и не сможете заблокировать IP адрес атакующего. Получив доступ к вашей конфигурации, злоумышленник может неделями подбирать пароль и вы об этом даже не узнаете.

Но данный брутфорс будет успешен, только если вы используете очень простой пароль. Попробуйте сгенерировать с помощью enable secret хэш для более сложного пароля и воспользуйтесь представленным выше сервисом (либо утилитой Cain&Abel). В этом случае на подбор пароля могут уйти годы.

Какой вывод можно сделать? Как написано на картинке выше “Don't use stupidly simple passwords”. Думаю перевод не требуется. О том, как выбирать пароли мы поговорим чуть позже.

Не используйте *enable password* даже в связке с *service password-encryption*. Используйте сложные пароли и *enable secret*.

Важное замечание, если у вас в системе используется и пароль enable password, и enable secret, то последний будет иметь более высокий приоритет (т.е. при входе в привилегированный режим будет использоваться пароль заданный с помощью enable secret).

1.1.3 Создание пользователей

Мы разобрались каким образом задавать пароль на вход в привилегированный режим. Но этого недостаточно. Нам также необходимо защитить и пользовательский режим, как это было описано выше. Для этого требуется настройка аутентификации по учетной записи. В этом случае при входе на устройство (даже с консоли) необходимо ввести логин (login) и пароль. Пример:

User Access Verification

Username: admin

Password:

Switch>

Для настройки учетных записей существует два основных способа:

1. Использование локальной базы пользователей. В этом случае учетные записи создаются непосредственно на устройстве и хранятся в его памяти.
2. Использование AAA-серверов. Все учетные записи хранятся на выделенном сервере. При этом нет необходимости создавать пользователей на сетевом оборудовании.

Я категорически не рекомендую использовать первый способ. Он менее безопасен, менее гибок и требует значительного внимания со стороны администратора. Использование локальной базы затрудняет соблюдение парольной политики (о которой мы поговорим чуть позже). Только представьте, что у вас около 30 сетевых устройств. И в компании три администратора с разными правами. Вам придется зайти на каждое устройство и вручную “вбить” учетные записи. Когда придет время менять пароли (а их нужно менять!), то данную процедуру придется повторить. Поэтому я настоятельно рекомендую использовать AAA-сервера (о них мы поговорим в следующей главе).

Но и не стоит забывать об адекватности применяемых решений. Как правило AAA-сервер рекомендуется если у вас больше 10 устройств. Если в вашей сети всего 3 - 5 устройств, то установка AAA-сервера будет выглядеть немного странно. Поэтому мы рассмотрим некоторые аспекты использования локальной базы пользователей.

Удивительно сколько внимания уделяется сложности паролей, но при этом все забывают про еще один параметр учетной записи - имя пользователя.

НИКОГДА не используйте стандартные учетные записи. Придумывайте уникальные имена пользователей.

Пример типичных учетных записей: admin, adm, administrator, root, cisco, user, usr и т.д. При брутфорсе почти всегда идет подбор паролей к стандартным именам. Задав уникальное имя учетной записи вы снижаете риск несанкционированного доступа во множество раз. Даже если вы будете использовать стандартный пароль вроде “cisco” или “1234” (хоть я это и не рекомендую), но имя учетной записи будет нестандартным, то злоумышленник не сможет получить доступ методом перебора паролей.

Теперь рассмотрим сам процесс создания учетных записей на устройствах Cisco. Как правило рекомендуют использовать команду:

```
Switch(config)#username admin privilege 15 password cisco
```

Никогда так не делайте. Мы уже знаем, что нельзя использовать стандартную учетную запись **admin**. Для задания пароля здесь используется **password**, недостатки которого мы уже обсудили выше. Используйте **secret**. Про пароль комментарии излишни. В этой команде есть еще один параметр, на который обычно не обращают внимания - **privilege**. Это уровень доступа. По умолчанию в Cisco IOS существует три уровня:

1. Уровень привилегий 0 (privilege 0). Это самый низкий уровень из которого доступны всего несколько команд: **disable**, **enable**, **exit**, **help** и **logout**. Используется редко.
2. Уровень привилегий 1 (privilege 1). Соответствует пользовательскому режиму (т.е. в качестве приглашения в командной строке **switch>**). Команды из привилегированного режима недоступны.
3. Уровень привилегий 15 (privilege 15). Привилегированный режим, где доступны все команды (приглашение в командной строке **switch#**).

Уровни 2-14 по умолчанию не используются, но команды, относящиеся к уровню 15, могут быть перенесены на один из этих уровней, также как и команды с уровня 1. Данная модель используется для разграничения пользователей в правах в зависимости от их роли. К примеру, администратору Интернет-провайдерской сети нужен доступ ко всем командам, поэтому ему понадобится наивысший уровень привилегий, т.е. 15. Специалистам тех. поддержки из первой линии возможно нужны только команды диагностики (вроде **ping** или **show mac-address-table**), без доступа к командам конфигурации. В таком случае подходит обычный пользовательский режим - уровень 1. Остальные уровни (2-14) можно использовать для создания кастомных (индивидуальных) профилей со строго определенным перечнем команд. Пример:

```
Switch(config)#username worker privilege 2 secret cisco/создаем пользователя
Switch(config)#privilege exec level 2 show running-config /определяем доступные команды
Switch(config)#privilege exec level 2 ping
```

Если зайти в систему под пользователем **worker**, то вам будет доступно всего две команды: **show running-config** и **ping**.

*Не создавайте учетные записи с уровнем привилегий 15. Пароль учетной записи и пароль на **enable** должны быть разными!*

Гораздо разумнее будет создавать пользователей с уровнем 1, тогда при входе на устройство вы будете попадать в пользовательский режим (**switch>**). Для перехода в привилегированный режим (**switch#**) будет необходимо дополнительно ввести команду **enable** и пароль. Это позволяет повысить уровень защищенности (злоумышленнику придется угадывать два пароля вместо одного). Таким образом создавая пользователя должна использоваться команда подобная примеру:

```
Switch(config)#username krok privilege 1 secret пароль
```

Здесь мы используем нестандартное имя учетной записи и пониженный уровень привилегий. Вопрос выбора паролей будет описан чуть позже.

При использовании AAA сервера учетные записи не хранятся на оборудовании, в этом случае можно присваивать уровень 15 (об этом мы поговорим чуть позже).

1.1.4 Подключение по консоли

Пароль на привилегированный режим задан, учетные записи созданы, пора обсудить настройку подключения. Начнем мы с консоли. Как было сказано выше, недостаточно задавать пароль только на привилегированный режим. Несанкционированный доступ к пользовательскому режиму также очень опасен. Поэтому мы настроим вход в консоль устройства по учетной записи, т.е. по логину и паролю. Чаще всего в интернете можно встретить следующую инструкцию:

```
Router#conf t           /вход в привилегированный режим
Router(config)#line console 0 /вход в настройки консоли
Router(config-line)#login local /указываем использование локальной базы пользователей
```

После этого при попытке входа в консоль устройства будет запрошен логин и пароль.

Однако это довольно старый способ и в любой современной литературе используют новый метод, который так и называется **new-model**. Настройка производится следующим образом:

```
Router(config)#aaa new-model           /указываем что будем использовать метод new-model
Router(config)#aaa authentication login default local /создаем method list
```

На этом настройка заканчивается. Если попробовать подключиться по консоли, то получим следующее:

User Access Verification

```
Username: admin
Password:
Router>
```

Теперь разберемся с использованными командами:

- **aaa new-model** включает функцию AAA (Authentication Authorization and Accounting) - систему аутентификации, авторизации и учета событий. Данная функция встроена в большинство версий IOS, однако по умолчанию она отключена. Преимущество **aaa new-model** в гибкости настройки аутентификации (в отличие от **login local**).
- **aaa authentication login** свидетельствует о настройке аутентификации. При этом создается так называемый метод аутентификации (**method list**).
- **default** - имя метода аутентификации. Для метода аутентификации всегда указывается имя (**list-name**). Это может быть либо default (как в нашем примере), либо другое конкретное имя. Таким образом, на разные типы линий (aux, vty, con...) можно назначить разные методы аутентификации, в зависимости от политики безопасности.
- **local** указывает на использование локальной базы пользователей.

В нашем примере используется **method list** под названием **default**, а это значит, что применяемые настройки касаются всех доступных линий на устройстве.

Таким образом мы одновременно настроили аутентификацию для консоли (**console**) и для удаленных подключений (**vty**).

Используя **aaa new-model** (вместо `login local`) в дальнейшем вы сможете более плавно перейти к использованию AAA-серверов, т.к. там используется именно этот метод. В главе AAA мы более подробно рассмотрим создание методов аутентификации.

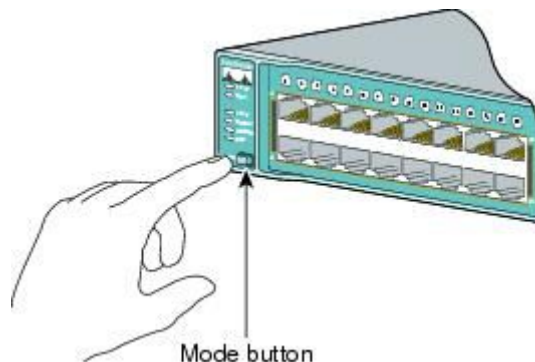
*Для настройки аутентификации используйте **aaa new-model**.*

Как ни странно, все еще встречаются прошивки, которые не поддерживают AAA. В этом случае ничего не остается, как использовать метод **login local** (либо обновить прошивку).

1.1.5 Сброс пароля

Последний параграф из секции “Физический доступ” касается возможности сброса пароля на устройстве. Казалось бы, вход в консоль устройства запаролен и можно не волноваться даже если злоумышленник получил физический доступ к оборудованию. Однако это ложное чувство безопасности. Все еще остается одна лазейка. Подключившись к устройству по консоли довольно просто установить новый пароль, даже не зная текущего. Вообще говоря, данная тема выходит за рамки книги, но мы обязаны рассмотреть сам процесс сброса, чтобы понять, насколько это просто сделать. Для примера рассмотрим алгоритм сброса пароля на коммутаторе:

- 1) Подключаемся по консоли к устройству (Мы рассмотрим пример с коммутатором. Для маршрутизатора процесс немного отличается, но идея та же.).
- 2) Вытаскиваем шнур питания коммутатора.
- 3) На лицевой панели коммутатора зажимаем кнопку **mode**.



- 4) Вставляем шнур питания не отпуская кнопку **mode**.
- 5) Не отпускаем кнопку **mode** до тех пор, пока индикатор над портом 1, не будет гореть, как минимум 2 секунды.
- 6) Вводим в консоли устройства команду **flash_init**.
- 7) Затем команду **load_helper**.
- 8) Переименуем текущую конфигурацию командой **rename flash:config.text flash:config.old**. Таким образом, при загрузке коммутатор не увидит файл с конфигурацией и запустится с заводскими настройками.
- 9) Загружаемся командой **boot**.

- 10) Входим в привилегированный режим командой **enable**. Вход будет без пароля.
- 11) Вернем название файла конфигурации: **rename flash:config.old flash:config.text**.
- 12) Теперь копируем первоначальную конфигурацию в текущую: **copy startup-config running-config**.
- 13) Мы оказались в привилегированном режиме с первоначальными конфигурациями. Для смены пароля необходимо войти в режим глобальной конфигурации: **conf t**.
- 14) Теперь мы можем делать с конфигурацией все, что угодно. Сменить пароль на enable, создать своего пользователя, настроить удаленное подключение и т.д.
- 15) Сохраняем конфигурацию: **copy running-config startup-config**.

Вообще говоря, процедура сброса может отличаться от устройства к устройству. Цель этого примера - показать наличие данной возможности. Как видим, получить полный контроль над коммутатором или маршрутизатором не составляет особого труда. При этом администратор сети может ничего не узнать, т.к. мы сохранили первоначальную конфигурацию и это не скажется на работе сети (за исключением времени, когда происходила перезагрузка коммутатора).

Чтобы избежать данной ситуации есть два варианта:

1. Ограничить физический доступ, как было сказано ранее. Сетевое оборудование должно быть либо в серверном помещении либо в специальном телекоммуникационном ящике с замком. К сожалению это не всегда возможно, особенно для коммутаторов уровня доступа, когда подключаются обычные пользователи.
2. Запретить возможность сброса пароля. Делается это на программном уровне используя команду по **service password-recovery**. Данная команда поддерживается в относительно новых прошивках. После ее применения уже не получится сбросить пароль с помощью описанной выше процедуры. Злоумышленник все еще сможет сбросить устройство в заводские настройки, но он не сможет получить доступ к текущей конфигурации. Однако подумайте несколько раз, прежде чем использовать эту функцию. Возможна ситуация, когда вы сами захотите сбросить пароль на устройстве (по причине его утери), но сделать это вы уже не сможете. Лично я рекомендую использовать данную функцию только на коммутаторах уровня доступа, находящихся вне серверной комнаты.

*Запретите сброс пароля (**no service password-recovery**) на коммутаторах уровня доступа к которым возможен физический доступ.*

1.2 Удаленный доступ

Просто поразительно насколько халатно относятся к удаленному доступу большинство системных администраторов. Защита удаленного доступа к оборудованию - один из самых важных пунктов в любой политике безопасности. Данный параграф будет посвящен основным аспектам настройки удаленного доступа

1.2.1 VTY

Прежде чем приступить к описанию методов защиты, необходимо рассмотреть сам процесс организации удаленного доступа. Если вам когда либо приходилось настраивать “удаленку”, то вы обязательно сталкивались с такой аббревиатурой как **VTY** - Virtual Teletype (хотя некоторые считают, что это расшифровывается как virtual terminal, а “y” достался по наследству от tty, применяемого в linux... однозначного ответа я так и не нашел). По-русски говоря это механизм позволяющий получить удаленный доступ к так называемому **command line interface (cli)**, т.е., по сути, к виртуальной консоли устройства. Для подключения к этой самой **vty** могут использоваться различные протоколы, о которых мы поговорим чуть позже.

Чаще всего в интернете можно встретить следующие инструкции по настройке **VTY**:

- 1) Вход по паролю

```
Router#conf t
Router(config)#line vty 0 4
Router(config-line)#password cisco
Router(config-line)#login
```

Как видим, в этом случае пароль задается непосредственно для VTY. Если попытаться подключиться с помощью telnet, то увидим следующее:

```
Router#telnet 192.168.1.1
Trying 192.168.1.1 ...Open
```

User Access Verification

Password:

Никогда не используйте этот метод!

- 2) Вход по учетной записи (username и password)

```
Router#conf t
Router(config)#line vty 0 4
Router(config-line)#login local
```

Здесь уже есть такой параметр как **local**, что означает использование локальной базы пользователей (т.е. у вас уже должен быть хотя бы один созданный **username**). При попытке удаленного доступа получим следующее:

```
Router#telnet 192.168.1.1
Trying 192.168.1.1 ...Open
```

User Access Verification

Username: admin

Password:

Router>

Если ваше устройство не поддерживает функцию **aaa new-model**, то этот способ более предпочтителен, чем первый, т.к. требуется ввести не только пароль, но и имя пользователя.

Однако оба метода это устаревший вариант настройки. Как было сказано выше, сейчас рекомендуется использовать **aaa new-model**. В параграфе по настройке консольного доступа (1.1.4) мы использовали следующие команды:

```
Router(config)#aaa new-model
Router(config)#aaa authentication login default local
```

В этом случае, благодаря параметру **default** мы настроили не только доступ по консоли, но и для **vty**. При этом мы указали, что нужно использовать локальную базу пользователей (**local**).

*Для настройки удаленного доступа используйте **aaa new-model**.*

Думаю у большинства читателей возникал вопрос, что это за цифры **0 4**? К тому же иногда встречаются **0 15**. Что это значит? Эти цифры означают всего лишь возможное количество удаленных подключений. Если мы используем **vty 0 4**, то создается 5 (0,1,2,3,4) виртуальных линий (консолей), к которым могут подключаться пользователи. Если к устройству одновременно попытаются подключиться 6 человек, то последний (шестой) не сможет этого сделать, т.к. все линии будут уже заняты. Используя команду **who** можно посмотреть кто подключен к устройству и к какой линии.

```
R3>who
      Line      User      Host(s)      Idle      Location
    0 con 0
  * 98 vty 0    steve      idle        00:00:00  10.0.0.1
```

Изначально оборудование cisco поддерживало именно 5 подключений. Но впоследствии эта цифра была увеличена до 16 (т.е. вариант **vty 0 15**). Если мы используем метод **aaa new-model**, то автоматически настраивается максимальное количество виртуальных линий. Если же вы используете **login local**, указав при этом параметр **line vty 0 15**, то после настройки в конфигурации можно увидеть следующее:

```
line vty 0 4
login local
line vty 5 15
login local
```

Т.е. 16 линий разделились на две группы: **0 4** и **5 15**. Зачем это делается, однозначного ответа нет. Скорее всего это сделано для некоего разграничения прав при удаленном подключении, однако на практике это редко используется. Как правило 5 виртуальных линий достаточно для любой организации. Дополнительные 11 линий (5 15), могут быть использованы для сторонних сервисов.

Следует заметить, что по умолчанию для удаленного подключения используется протокол **Telnet**. Кроме него еще существуют **SSH** и **HTTPS**. В следующих параграфах мы рассмотрим их более подробно.

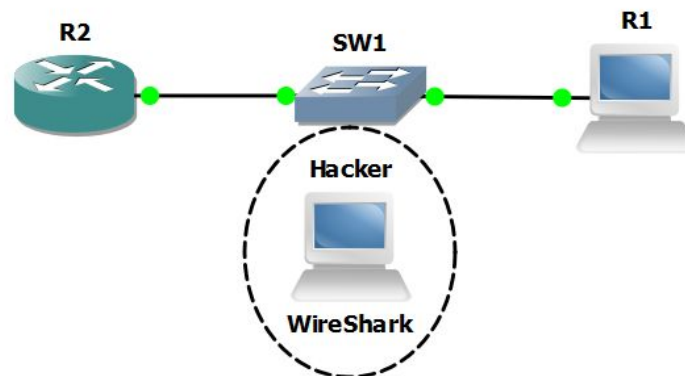
Также не стоит забывать про таймаут сессии, т.е. время, в течении которого подключение будет активным, даже если не происходит никаких действий:

```
Router(config-line)#exec-timeout 5 0
```

В данном случае мы установили таймаут сессии в 5 минут и 0 секунд. Эта команда весьма полезна если вы подключились к устройству, а потом “пошли пить чай”. Сессия автоматически прервется через 5 минут и никто не сможет ей воспользоваться без вашего ведома.

1.2.2 Telnet

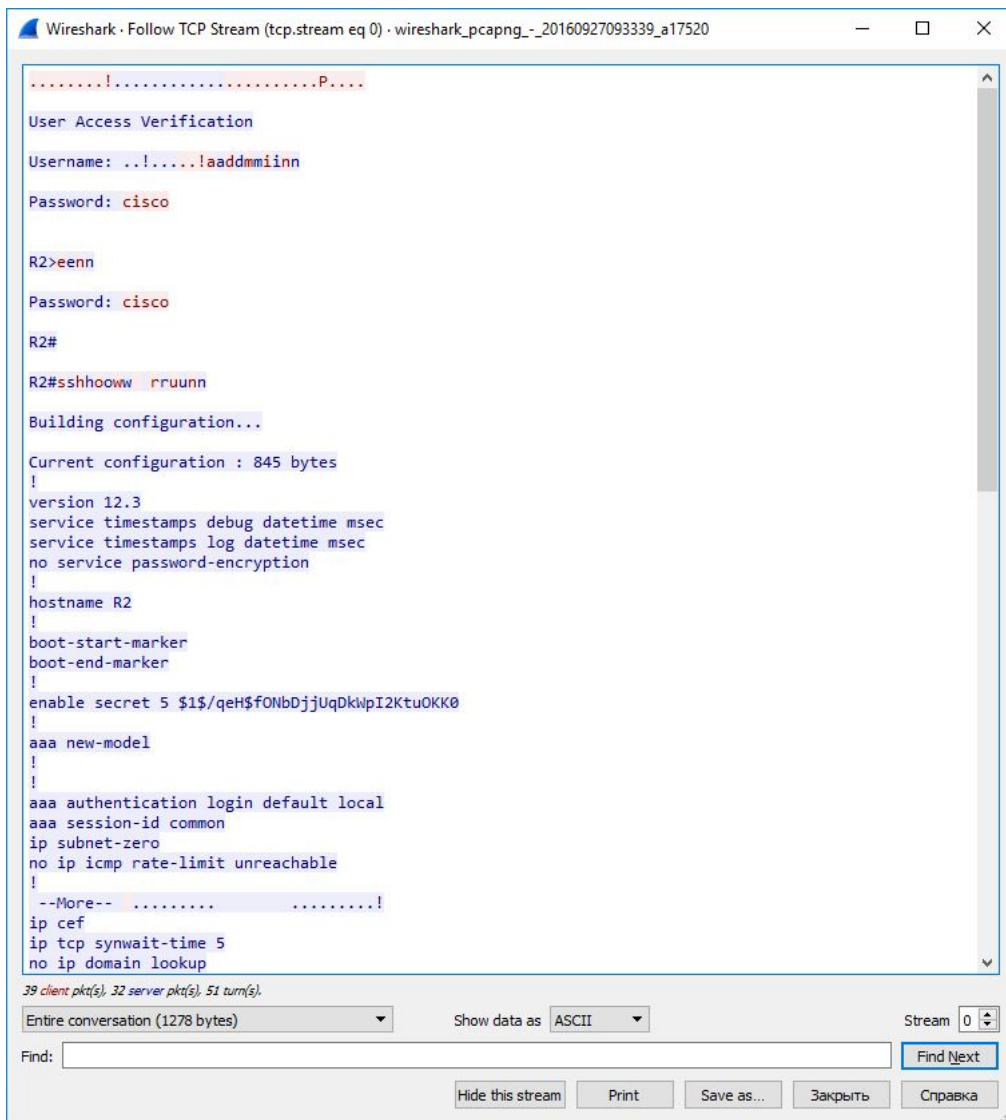
Telnet. Этот протокол доступен практически на любом сетевом оборудовании и почти всегда используется по умолчанию. Имеет ярко выраженный недостаток - все данные передаются в открытом виде. Давайте рассмотрим простейший пример (вы можете его с легкостью повторить в GNS3 или UNetLab). Имеется следующая схема:



Предположим, что хакер получил доступ к коммутатору **SW1**. Либо он мог поставить свой коммутатор в разрыв существующей линии. Настроил зеркалирование трафика (**SPAN порт**) на свой компьютер, где в свою очередь запущен **Wireshark** - софт для анализа трафика. Администратор ни о чем не подозревает и инициирует удаленное telnet соединение с роутером **R2**. Вводит имя, пароль, просматривает конфигурацию. В это время на компьютере злоумышленника видно весь проходящий трафик, в том числе **TELNET**:

19	32.542761	192.168.1.2	192.168.1.1	TELNET	63 Telnet Data ...
20	32.542761	192.168.1.2	192.168.1.1	TCP	60 [TCP Dup ACK 18#1
22	32.552269	192.168.1.1	192.168.1.2	TELNET	66 Telnet Data ...
23	32.556272	192.168.1.2	192.168.1.1	TELNET	60 Telnet Data ...
24	32.556772	192.168.1.2	192.168.1.1	TELNET	60 Telnet Data ...
25	32.557272	192.168.1.2	192.168.1.1	TELNET	63 Telnet Data ...
26	32.565810	192.168.1.1	192.168.1.2	TELNET	96 Telnet Data ...
27	32.566312	192.168.1.1	192.168.1.2	TELNET	60 Telnet Data ...
28	32.566812	192.168.1.1	192.168.1.2	TELNET	60 Telnet Data ...
29	32.568327	192.168.1.2	192.168.1.1	TELNET	60 Telnet Data ...

Если воспользоваться встроенной функцией Wireshark, а именно **Analyze->Follow->TCP Stream**, то можно получить следующую картину:



Как видно из картинки, злоумышленник теперь знает и логин, и пароль, и даже всю конфигурацию устройства. Сказывается отсутствие шифрования Telnet - сессии.

*Не используйте **Telnet** в качестве протокола удаленного подключения.*

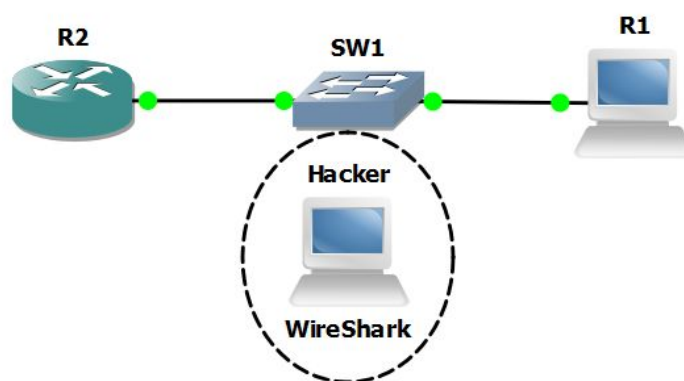
1.2.3 SSH

Очевидное решение проблемы Telnet - шифрование данных. Для этой цели был создан протокол **SSH (Secure Shell** - "Безопасная оболочка") - сетевой протокол прикладного уровня, предназначенный для удаленного управления. SSH является практически стандартом для удаленного администрирования и поддерживается большинством операционных систем. Протокол создает виртуальный зашифрованный канал между двух устройств. При этом одно устройство выступает в качестве **SSH сервера** (например маршрутизатор или коммутатора), а другое - в качестве **SSH клиента** (например наш компьютер). Все данные внутри канала шифруются, в том числе login и password. Вообще говоря, SSH может использоваться не только для

удаленного управления, но и для безопасной передачи любых данных, будь-то звук, видео и т.д.

Следует отметить, что существует две версии протокола, это **SSHv1** и **SSHv2**. Как можно догадаться, предпочтительнее использовать именно SSHv2, он более безопасен. Эта безопасность достигается за счет использования более криптостойких алгоритмов шифрования **3DES** или **AES**. При этом следует учитывать, что некоторое оборудование по умолчанию не поддерживает данные алгоритмы, а значит возможно использовать только SSHv1. Эта проблема решается банальным обновлением прошивки. Как правило это прошивка в названии которой присутствуют символы **"k9"**.

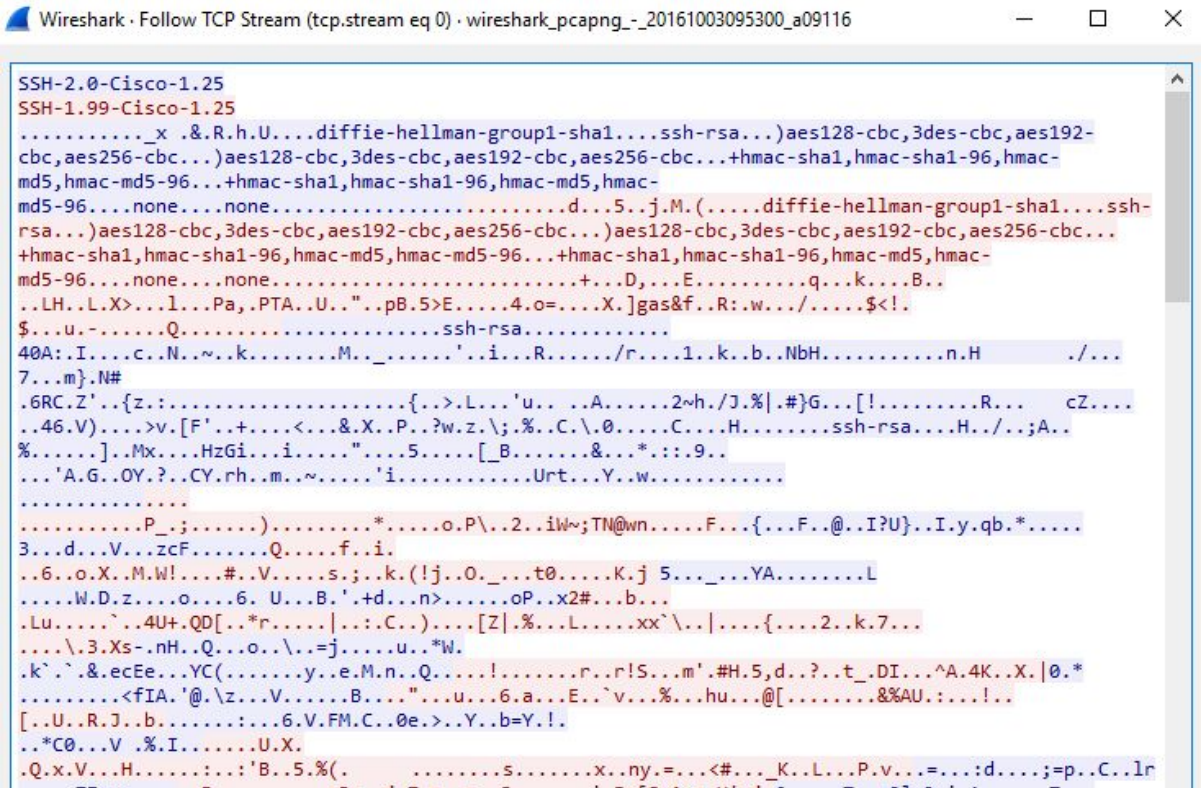
Вернемся к описанному ранее примеру:



Предположим, что теперь администратор использует SSH для подключения к маршрутизатору **R2**. Он подключается с помощью **Putty** (либо **SecureCRT**, это не важно), вводит пароль, просматривает конфигурацию. Злоумышленник, как и в первом примере, видит весь трафик и использует **Wireshark** для анализа пакетов. Если посмотреть “дамп” трафика, то можно увидеть следующее:

28	19.692660	192.168.1.2	192.168.1.1	SSHv2	60 Encrypted packet (len=4)[Malformed
29	19.700566	192.168.1.1	192.168.1.2	SSHv2	122 Server: Encrypted packet (len=68)
30	19.703092	192.168.1.2	192.168.1.1	SSHv2	118 Client: Encrypted packet (len=64)
31	19.713117	192.168.1.2	192.168.1.1	SSHv2	90 Client: Encrypted packet (len=36)
32	19.720645	192.168.1.1	192.168.1.2	SSHv2	122 Server: Encrypted packet (len=68)
33	19.938436	192.168.1.2	192.168.1.1	TCP	60 19115 → 22 [ACK] Seq=681 Ack=952 W
34	23.666569	192.168.1.2	192.168.1.1	SSHv2	106 Client: Encrypted packet (len=52)
35	23.678590	192.168.1.1	192.168.1.2	SSHv2	90 Server: Encrypted packet (len=36)
36	23.686602	192.168.1.2	192.168.1.1	SSHv2	118 Client: Encrypted packet (len=64)
37	23.697052	192.168.1.2	192.168.1.1	SSHv2	60 Encrypted packet (len=4)[Malformed
38	23.699051	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)
39	23.707338	192.168.1.2	192.168.1.1	SSHv2	118 Client: Encrypted packet (len=64)
40	23.717354	192.168.1.2	192.168.1.1	SSHv2	74 Encrypted packet (len=20)[Malformed
41	23.719870	192.168.1.1	192.168.1.2	SSHv2	90 Server: Encrypted packet (len=36)
42	23.727378	192.168.1.2	192.168.1.1	SSHv2	106 Client: Encrypted packet (len=52)
43	23.730400	192.168.1.1	192.168.1.2	SSHv2	90 Server: Encrypted packet (len=36)
44	23.742083	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)
45	24.114932	192.168.1.2	192.168.1.1	TCP	60 19115 → 22 [ACK] Seq=937 Ack=1164
46	26.378067	192.168.1.2	192.168.1.1	SSHv2	106 Client: Encrypted packet (len=52)
47	26.390084	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)
48	26.402085	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)
49	26.412592	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)
50	26.423300	192.168.1.1	192.168.1.2	SSHv2	106 Server: Encrypted packet (len=52)

Можно заметить, что используется протокол **SSHv2** и все пакеты в зашифрованном виде - **Encrypted**. Если снова воспользоваться утилитой **Analyze->Follow->TCP Stream**, то мы увидим следующее:



В данном случае мы уже не можем увидеть ни пароль, ни конфигурацию, что делает нецелесообразным “прослушку” управляющего трафика.

 Используйте защищенные протоколы удаленного подключения (например **SSHv2**).

Давайте теперь вкратце рассмотрим процесс настройки SSH подключений, тем более что это довольно простой процесс. Большую часть настроек мы уже выполнили в предыдущих этапах. На всякий случай напомним, что у вас уже должен быть задан пароль для **enable** и создан хотя бы один **username**. Затем настроена аутентификация либо с помощью **aaa new-model** (aaa authentication login default local) либо **login local**, для настройки SSH нет никакой разницы, т.к. в данном случае мы всего лишь выбираем протокол. Вот так выглядит процесс настройки:

```
R2(config)#ip domain-name netskills.ru           /имя домена используется для генерации ключей
R2(config)#crypto key generate rsa modulus 1024  /генерируем пару ключей, где длина ключа - 1024
```

Длина ключа может варьироваться от 360 до 2048. После ввода команды маршрутизатор сообщит об успешной генерации ключей и о том, что был включен SSH:

```
The name for the keys will be: R1.netskills.ru

% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
*Mar 1 00:01:54.243: %SSH-5-ENABLED: SSH 1.99 has been enabled
```

Теперь укажем, что нужно использовать SSH версии 2:

```
R2(config)#ip ssh version 2
```

Воспользуемся дополнительными настройками:

```
R2(config)#ip ssh time-out 15
R2(config)#ip ssh logging events
```

/Ограничиваем время ввода логина/пароля
/Логируем все попытки входа

И последнее, что нам нужно сделать, это указать SSH в качестве протокола удаленного доступа:

```
R2(config)#line vty 0 4
R2(config-line)#transport input ssh
```

Обратите внимание, что если в конфигурации устройства присутствуют две группы (**vtty 0 4** и **vtty 5 15**), то команду **transport input ssh** нужно указать дважды.

По завершении настройки обязательно проверьте возможность подключения по **Telnet**, доступ должен быть запрещен.

1.2.4 HTTP/S

Кроме Telnet и SSH, коммутаторы и маршрутизаторы Cisco поддерживают еще один вариант управления - через веб-интерфейс. Могут использоваться такие протоколы, как **HTTP** и **HTTPS**. Пример веб-интерфейса представлен на картинке ниже.

Cisco Systems

Accessing Cisco 3725 "R1"

[Show diagnostic log](#) - display the diagnostic log.

[Monitor the router](#) - HTML access to the command line interface at level [0](#)[1](#)[2](#)[3](#)[4](#)[5](#)[6](#)[7](#)[8](#)[9](#)[10](#)[11](#)[12](#)[13](#)[14](#)[15](#)

[Show tech-support](#) - display information commonly needed by tech support.

[Extended Ping](#) - Send extended ping commands.

[QoS Device Manager](#) - Configure and monitor QoS through the web interface.

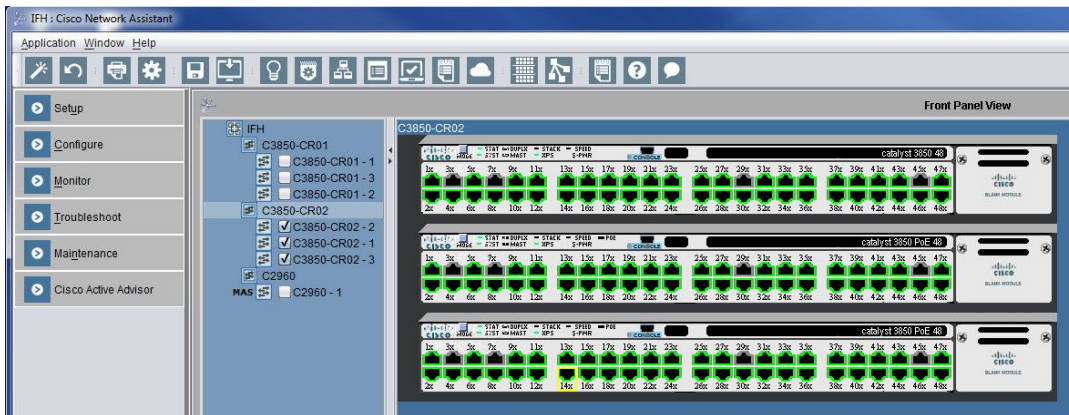
Help resources

1. [CCO at www.cisco.com](#) - Cisco Connection Online, including the Technical Assistance Center (TAC).
2. [tac@cisco.com](#) - e-mail the TAC.
3. 1-800-553-2447 or +1-408-526-7209 - phone the TAC.
4. [cs-html@cisco.com](#) - e-mail the HTML interface development group.

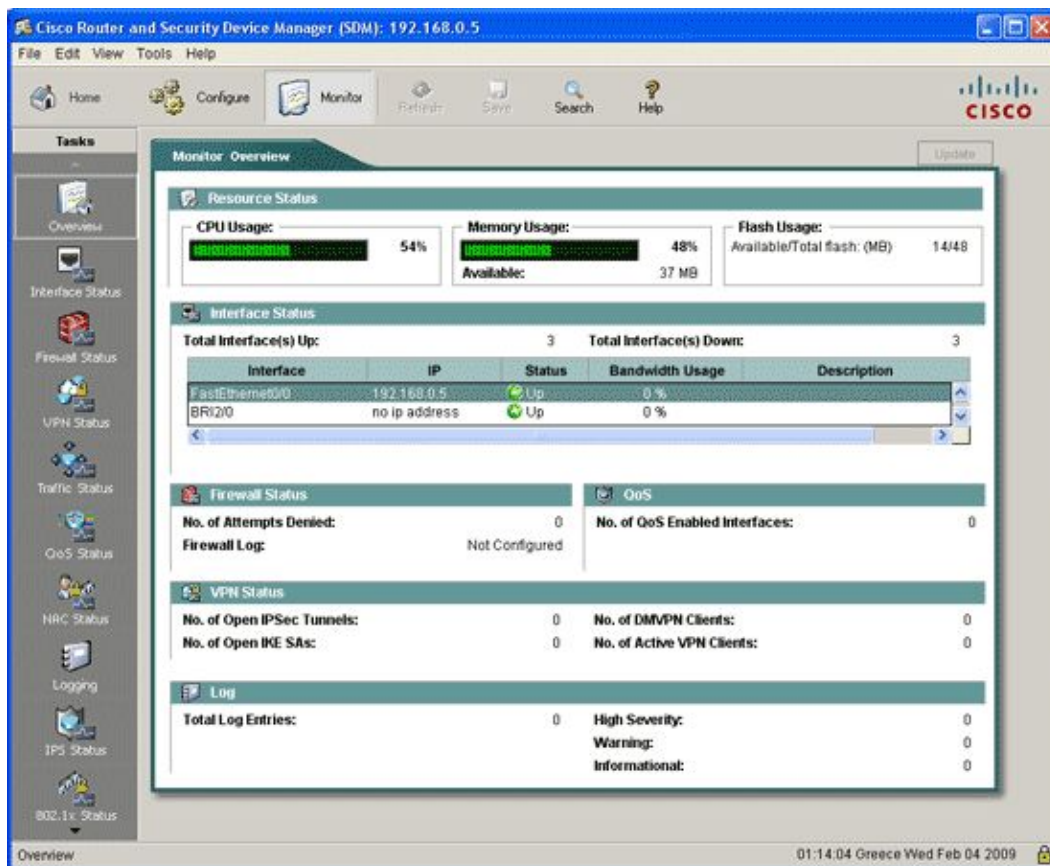
Многие сетевые инженеры даже не знают о такой возможности по причине довольно удобной командной строки Cisco.

Для реализации управления через веб-интерфейс на устройстве используется **локальный http/https сервер**. На самом деле, практически никто не использует веб-интерфейс для управления устройством. **HTTP/S** сервер включают только ради использования специализированных программ (с графическим интерфейсом) управления устройствами Cisco. Наиболее популярны следующие программы:

- 1) **CNA - Cisco Network Assistant**. Программа с графическим интерфейсом которая существенно упрощает администрирование устройств Cisco (коммутаторы и маршрутизаторы) и автоматизирует выполнение рутинных задач. Программа позволяет управлять сетевой инфраструктурой, которая включает до 40 устройств.



- 2) **SDM - Security Device Manager.** Данная программа позволяет управлять маршрутизаторами Cisco. Существенно упрощает создание VPN туннелей, а также предоставляет более удобную работу со списками доступа (access-list).



Подробное рассмотрение данных программ выходит за рамки нашей книги. Однако мы не можем не затронуть проблему безопасности, связанную с использованием HTTP/S сервера. В первую очередь стоит отметить, что именно с HTTP/S серверами связано наибольшее количество уязвимостей, позволяющих получить удаленный контроль над устройством. К тому же, большинство вендоров весьма халатно относятся к устранению этих самых уязвимостей, поскольку не считают эту задачу приоритетной. Оно и неудивительно, как я сказал ранее, мало кто пользуется данной функцией. В итоге получается, что ваше устройство может месяцами “святиться” в сети с набором известных уязвимостей. Используйте данную

функцию весьма осторожно и следите за обновлением прошивок. Мы поговорим о прошивках чуть подробнее в следующих главах.

Как ни странно, но на некоторых устройствах функция HTTP включена по умолчанию, даже если вы ее не используете. Обязательно проверьте это, либо выполните команду **no ip http server** в режиме глобальной конфигурации (**conf t**).

Если же по какой-либо причине вам все же необходимо использовать web доступ (например для CNA или SDM), то выбирайте **HTTPS**. Здесь можно провести аналогию и сравнить HTTP с Telnet, а HTTPS с SSH. HTTPS шифрует все данные, в то время как HTTP передает их в открытом виде, в том числе и пароли.

Давайте рассмотрим сам процесс настройки:

```
R1(config)#ip domain-name netskills.ru      /должно быть задано доменное имя
R1(config)#ip http secure-server           /включаем https сервер
R1(config)#no ip http server               /убедитесь, что http сервер выключен
```

На этом настройка закончена. Т.к. мы ранее использовали **aaa new-model** и команду **aaa authentication login default local**, то аутентификация уже настроена и вход будет выполняться по созданным локальным учетным записям (**username**). Для подключения к устройству используйте браузер с запросом **https://ip-address-router**, либо озвученные ранее программы (CNA, SDM).

При необходимости организации web-доступа к оборудованию, используйте HTTPS вместо HTTP. Помните об огромном количестве уязвимостей, связанных с HTTP/S.

1.2.5 Ограничение доступа

В 90% организаций, где мне приходилось работать, к сетевому оборудованию был разрешен удаленный доступ абсолютно с любого компьютера. Только подумайте, насколько повышается риск несанкционированного доступа к коммутатору или маршрутизатору, если доступ возможен не с одного - двух компьютеров, а с тысячи? Для повышения безопасности, логичным выходом является ограничение доступа к оборудованию. Есть два основных способа:

1) Out-of-band (OOB) - управление сетью по выделенному каналу. В этом случае управляющий трафик изолируется от общего (пользовательского). При этом под управляющим трафиком может пониматься не только SSH или Telnet сессия, но и такой трафик как SNMP и Log (мы поговорим об этом в следующих главах). Есть два способа отделить управляющий трафик от общего:

а) Физически. Устройства, которыми необходимо управлять, объединяются в выделенную физическую сеть. Для этого на маршрутизаторах и коммутаторах, как правило, имеется специальный порт - **management port (mgmt)**. Если такового не имеется, то можно в произвольном порядке выделить порт под эти нужды. Таким образом, управлять оборудованием сможет только тот компьютер, который физически находится в управляющей сети. Это может быть специально выделенный компьютер системного администратора. Данный способ обладает наибольшей безопасностью, но весьма редко применяется. Связано это в первую очередь со сложностью исполнения, т.к. приходится создавать отдельную сеть для всех устройств, а это либо дорого (тянуть

дополнительные линии связи, устанавливая дополнительный коммутатор), либо просто невозможно (в случае большой распределенной сети).

б) Логически. В этом случае управляющий трафик отделяется от пользовательского посредством выделения в отдельный логический сегмент - **VLAN**. Данный способ является компромиссным решением, т.к. мы изолируем управляющий трафик без необходимости в дополнительных линиях связи. Получить удаленный доступ к оборудованию сможет только тот компьютер, которых находится в управляющем VLAN. Логическое изолирование также не применимо в случае распределенной сети, если при этом невозможно “прокинуть” управляющий VLAN до удаленного объекта.

Оба варианта требуют настройки на оборудовании IP - адреса из сети, отличной от сети пользователей.

- 2) In-band (IB)** - управление оборудованием осуществляется по общим каналам. Данный способ используется чаще всего, либо из-за отсутствия возможности организации способа **OOB**, либо из-за обычной халатности системного администратора. В этом случае значительно повышается вероятность несанкционированного доступа (или попытки доступа). Для повышения уровня защищенности единственным разумным решением является применение списков доступа (**Access-list**). Мы поговорим о них более подробно в следующем параграфе.

Ограничьте удаленный доступ. Лишь несколько компьютеров должны иметь возможность подключения к оборудованию.

1.2.6 Списки доступа к оборудованию

Если у вас нет возможности вынести управляющий трафик в отдельный VLAN, то практически единственным средством ограничения доступа являются списки доступа - **Access List**. Используя их вы сможете указать с каких именно компьютеров возможно управление устройствами, т.е. подключение по Telnet, SSH или HTTPS. Тем самым вы отсечете саму возможность несанкционированного доступа с запрещенных узлов, в том числе сети Интернет (на самом деле все еще остается лазейка в виде спуфинга, о которой мы поговорим чуть позже).

Первое что необходимо сделать, это четко определить с каких компьютеров будет возможно удаленное подключение к устройствам. Как правило это компьютер системного администратора. Разумеется IP-адрес этого компьютера должен быть статическим. При этом, если вы используете в сети DHCP сервер, следует заранее зарезервировать данный адрес, чтобы он не был случайно выдан другому пользователю.

Также я настоятельно не рекомендую настраивать удаленное подключение из сети Интернет, даже если вы используете защищенные протоколы вроде SSH. Гораздо логичнее будет организация **VPN соединения**, где компьютеру удаленного пользователя будет присваиваться некий зарезервированный IP-адрес и уже с этого адреса будет возможен доступ к оборудованию.

Сам процесс настройки списков доступа предельно прост. Давайте рассмотрим ограничение доступа по SSH (при этом у вас уже должны быть выполнены предыдущие настройки):

```
R1(config)#ip access-list standard SSH-ACCESS      /создание списка доступа с понятным названием
R1(config-std-nacl)#permit host 192.168.2.2      /определяем хосты имеющие доступ по SSH
R1(config-std-nacl)#permit host 192.168.2.3
R1(config-std-nacl)#exit
R1(config)#line vty 0 4                          /заходим в режим конфигурации vty
R1(config-line)#access-class SSH-ACCESS in       /"вешаем" список доступа
```

Теперь доступ по SSH возможен только с двух узлов - 192.168.2.2 и 192.168.2.3. При этом если нужно указать целую сеть, то можно использовать команду **permit 192.168.2.0**.

Функция **ip access-list** (именованные списки доступа) может быть недоступна на устройствах со старой прошивкой. Тогда необходимо либо обновить прошивку, либо использовать следующий вариант настройки:

```
R1(config)#access-list 1 remark SSH-ACCESS
R1(config)#access-list 1 permit host 192.168.2.2
R1(config-std-nacl)#exit
R1(config)#line vty 0 4
R1(config-line)#access-class 1 in
```

Если вы используете HTTPS на ваших устройствах, то ограничение доступа можно выполнить используя тот же список доступа следующим образом:

```
R1(config)# ip http access-class 1
```

1.2.7 Защита от Brute Force

Кроме упомянутых выше методов (выбор сложного пароля, нестандартной учетной записи и списков доступа) есть еще один способ, который позволит защититься от попытки взломать ваше устройство. Это так называемая защита от Brute Force (т.е. от перебора паролей).

Здесь я хотел бы сделать небольшое отступление и вспомнить один случай из жизни. Пару лет назад я помогал своему другу настроить Mikrotik в качестве пограничного маршрутизатора (т.е. он использовался для выхода в Интернет). Прделав базовые настройки мы организовали выход в Интернет буквально за 10 минут. После мы взяли по бутылочке пива и я в расслабленном режиме принялся ему рассказывать про Mikrotik, как его администрировать, где и что настраивать. Когда мы дошли до функции просмотра логов, то заметили странную вещь. Из внешней сети постоянно кто-то пытался подключиться к нашей "железке". При этом в попытках использовались такие учетные записи как "root", "admin", "guest" и так далее (т.е. стандартные учетки). Посмотрев IP - адреса атакующих мы выяснили, что эта активность ведется из Китая. Но кому нужен наш Mikrotik? Немного "погуглив" и почитав форумы, мы поняли, что это обычные китайские боты, которые в автоматическом режиме по всему миру ищут уязвимые устройства и выполняют "атаку на дурака". Естественно для них не представляет интерес какая-то наша личная информация (хотя кто знает?). С большой вероятностью им просто нужны взломанные устройства с помощью которых они смогут осуществлять массированные DDoS атаки

(чуть позже мы обсудим этот тип атак). Именно поэтому не пренебрегайте ранее данными советами:

- 1) Не используйте стандартные имена в учетных записях;
- 2) Придумывайте сложные пароли (мы обсудим каким образом выбирать пароль в следующей главе);
- 3) Ограничьте удаленный доступ с помощью списков доступа (для сети Интернет его лучше вообще выключить).

Однако бывают экзотические случаи, когда все же необходим доступ к оборудованию из сети Интернет. При этом нет возможности указать конкретный удаленный хост, с которого будет разрешен доступ. Да и кто даст гарантию, что внутри вашей сети нет зараженного компьютера, который будет пытаться подобрать пароли ко всем доступным хостам? Именно для таких случаев есть дополнительный уровень защиты - задержка между повторными вводами учетных данных. Настраивается это следующим образом:

```
R1(config)#login delay 5
R1(config)#login block-for 60 attempts 3 within 30
```

Первой командой мы указываем интервал в 5 секунд между повторными вводами паролей. Второй командой мы настраиваем блокировку входа на 60 секунд, если в течении 30 секунд было 3 попытки неудачного входа. Данный способ существенно ограничивает злоумышленников при подборе паролей. Получается, что они смогут подбирать всего 3 пароля в течении 15 секунд, а затем блокировка на 60 секунд. Даже если вы используете довольно простой пароль, у злоумышленников уйдет очень много времени на подбор, что становится совершенно нецелесообразно. При этом в консоли устройства вы увидите сообщение примерно следующего содержания:

```
*Mar 1 00:04:55.899: %SEC_LOGIN-1-QUIET_MODE_ON: Still timeleft for watching failures is 4 secs, [user: admin]
[Source: 192.168.10.3] [localport: 23] [Reason: Login Authentication Failed] [ACL: sl_def_acl] at 00:04:55 UTC Fri Mar 1
2002
R1#
*Mar 1 00:05:03.491: %SEC-6-IPACCESSLOGP: list sl_def_acl denied tcp 192.168.10.3(62741) -> 0.0.0.0(23), 1 packet
```

Т.е. при 3-х неудачных попытках создается временный access-list, который блокирует дальнейшие попытки с определенного хоста. Максимальное время блокировки - 65535 секунд (т.е. примерно на 18 часов). Крайне не рекомендую выставлять такой интервал, т.к. вы и сами можете быть заблокированы если вдруг сделаете несколько неудачных попыток (перепутали пароль, русская раскладка клавиатуры, зажатый caps lock).

Для защиты от подбора паролей (Brute Force) настройте временную блокировку повторной аутентификации.

1.2.8 Идентификация устройства

Почти уверен, что в работе каждого сетевого администратора были моменты когда он нечаянно вводил команду “не на том” коммутаторе/маршрутизаторе после чего падала сеть, либо пропадал доступ к оборудованию. Наверняка многие сталкивались с ситуацией, когда зайдя на устройство было трудно понять куда именно вы попали. Все это также является аспектами безопасности вашей сети. Как вы уже

наверно поняли, здесь пойдет речь о таких вещах, как **hostname** и **login banner**. Довольно часто про эти опции просто забывают.

Вообще говоря, многие начинают настройку устройства с имени - **hostname**. Это уже дело привычки. Имя должно быть осмысленным и однозначно определять предназначение устройства. Пример:

- Коммутатор ядра - **4507_Core**
- Коммутатор доступа на втором этаже - **2960_2flor**
- Пограничный маршрутизатор - **1710_Edge**

Это лишь примеры, которые отражают суть идеи. Естественно, что у каждого могут быть собственные стандарты. Сама настройка элементарна:

```
Router>enable
Router#conf t
Router(config)# hostname R1           /задаем имя
R1(config)#                          /имя изменено
```

Кроме имени есть еще одна полезная опция - **login banner**. Это сообщение которое мы видим при подключении к устройству. Пример:

```
User Access Verification
```

```
Username: admin
Password:
```

```
Test banner for netskills
```

```
R1>
```

Какую же информацию можно поместить в сообщение баннера? Как правило в нем указывают информацию об устройстве, о том, что нужно быть аккуратным при конфигурировании и о том, что нужно немедленно выйти если вы попали на это устройство случайно (такое все же иногда случается). Когда это может быть полезно? К примеру у вас несколько администраторов сети и не каждому позволено конфигурировать некоторые устройства. В этом случае, зайдя на устройство, можно однозначно увидеть предостережение и сделать вывод о “законности” дальнейшей настройки.

При этом у нас есть выбор когда показывать это сообщение - до аутентификации или уже после (как в примере выше). На мой взгляд гораздо логичнее это делать после, особенно если в сообщении содержится важная информация (например, что данный коммутатор является ядром сети). Настройка также элементарна:

```
R1(config)#banner exec c                /начало баннера
Enter TEXT message. End with the character 'c'.
Test banner for NetSkills                /сообщение
c                                        /конец баннера
R1(config)#
```

Теперь при аутентификации на устройстве вы увидите сообщение “**Test banner for NetSkills**”. Если же есть необходимость использовать баннер до аутентификации, то используйте команду **banner login** вместо **banner exec**.

*Задавайте имя устройства (**hostname**) и сообщение после аутентификации (**banner exec**), чтобы в будущем не “перепутать” оборудование.*

1.3 AAA

Как уже было сказано выше, **AAA-сервер** позволяет централизованно хранить все учетные записи пользователей. Это отличная альтернатива локальной базе на самих устройствах. Какие же преимущества несет в себе AAA-сервер:

- 1) Удобное администрирование учетных записей. Если у вас больше 10-и сетевых устройств, то добавление даже одного пользователя превращается в весьма нудную и долгую процедуру. AAA-сервер решает эту проблему.
- 2) Безопасность учетных записей. В случае с локальной базой пользователей (login local), при несанкционированном доступе к устройству, злоумышленник потенциально получает доступ ко всем паролям, которые как правило одинаковы для всех коммутаторов и маршрутизаторов. При использовании AAA-сервера, все пароли хранятся централизованно, а на устройствах отсутствуют какие-либо записи. Здесь есть и обратная сторона медали. Если злоумышленник сможет получить доступ к AAA-серверу, то вся сеть окажется скомпрометирована. Поэтому уделяйте особое внимание защите AAA-серверов.
- 3) Простота соблюдения парольной политики. Как я уже говорил ранее, пароли должны обязательно меняться с определенной периодичностью. В этом случае AAA-сервер является незаменимым помощником. Смена паролей занимает считанные секунды, даже если в вашей сети сотни устройств. Что такое “парольная политика” мы подробно рассмотрим в следующих главах.

*Если в вашей сети более 10-и устройств, то вы просто обязаны использовать
AAA-сервер.*

Но обо всем по порядку. Что же значат эти три буквы? Расшифруем:

A - Authentication (аутентификация)

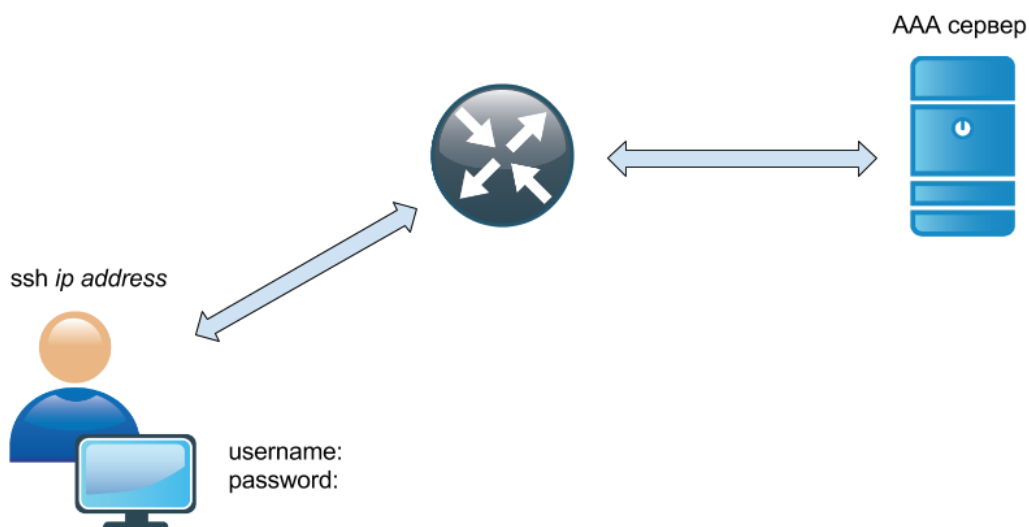
A - Authorization (авторизация)

A - Accounting (учет)

Буквы расшифровали, но это не сильно прояснило смысл этой технологии. Для разъяснений я воспользуюсь примером моего любимого коуча - Keith Barker из СВТ Nuggets. Представим, что вы пришли в банк, чтобы снять часть своих сбережений. Для начала вас попросят предоставить документы, чтобы убедиться, что вы являетесь клиентом банка. После того как вы показали документы, вы прошли аутентификацию (**authentication**), если конечно вы являетесь клиентом банка. Далее, девушка, которая вас обслуживает, проверяет, можете ли вы снимать деньги и снимать именно такую сумму. Если все нормально, то вы прошли авторизацию (**authorization**). А после всех этих действий, естественно ведутся записи в журнал, что вы приходили, сняли деньги, на счету осталось столько то денег. В данном случае прошел учет (**accounting**). Надеюсь теперь вам более понятно, что такое AAA и для чего это используется.

У многих возникают сложности при произношении “AAA”. Как же правильно произносить имя этой технологии? Чаще всего используется фраза “**трипл эй**” (triple A), т.е. “тройное А”.

Теперь рассмотрим процесс AAA на примере сетей. Представим, что на сетевом устройстве уже настроен AAA и в нашей сети есть AAA-сервер с настроенными учетными записями. В данном случае маршрутизатор выступает в роли AAA-клиента.



Распишем процесс поэтапно (естественно в упрощенном виде.)

- 1) Пользователь пытается подключиться к маршрутизатору по telnet или ssh.
- 2) Маршрутизатор запрашивает у пользователя учетные данные - username и password.
- 3) Пользователь вводит свои учетные данные.
- 4) Маршрутизатор обращается к серверу AAA с вопросом: "Разрешать доступ этому пользователю?". И передает введенные данные.
- 5) AAA-сервер ищет учетные данные пользователя. Находит (или нет) их. В данном случае пользователь прошел (или не прошел) аутентификацию (authentication).
- 6) AAA-сервер также находит уровень привилегий для данного пользователя, который определяет его права. Это уже авторизация (authorization).
- 7) Маршрутизатору дается ответ: "Можно пустить этого пользователя с уровнем привилегий X", где X - от 0 до 15.
- 8) Маршрутизатор разрешает пользователю доступ к командной строке с определенным уровнем привилегий.
- 9) Естественно, что данные операции логируются в журнале событий, т.е. происходит учет (accounting).

Данное общение (между AAA-клиентом и AAA-сервером) выполняется при помощи **AAA-протокола**. В настоящее время наибольшее распространение получили два протокола: **RADIUS** и **TACACS+**. Далее мы рассмотрим их основные отличия.

1.3.1 RADIUS

RADIUS (Remote Authentication in Dial-In User Service) - открытый AAA-протокол. Рассмотрим ключевые характеристики данного протокола:

- 1) В качестве транспортного протокола используется **UDP**. Для аутентификации порт 1812 (ранее 1645), а для учета - 1813 (ранее 1646).
- 2) При взаимодействии между AAA-сервером и AAA-клиентом, протокол шифрует только пароль. Остальные данные передается в открытом виде.
- 3) Аутентификация и авторизация являются единым процессом. Как только пользователь проходит аутентификацию, он тут же, автоматически проходит авторизацию и получает соответствующий ему уровень привилегий (возможность выполнять ту или иную команду). После этого, пользователь волен делать все, что ему разрешено в рамках своих привилегий.
- 4) Имеет очень детальный процесс учета (accounting).
- 5) Открытый протокол (open source).

Ярким представителем данного протокола является **FreeRADIUS** - AAA-сервер с открытым исходным кодом. Также может быть интересна реализация этого протокола с графическим интерфейсом - **daloRADIUS**.

1.3.2 TACACS+

TACACS+ (Terminal Access Controller Access Control System). Произносится как ТА-КАКС плюс. Это проприетарная разработка компании Cisco и является более усовершенствованным вариантом более старого протокола TACACS. Иногда можно услышать слово TACACS без приставки "+". Можете быть уверены, что на самом деле речь идет именно о TACACS+, поскольку протокол TACACS уже практически не используется. Основные отличия:

- 1) В качестве транспортного протокола используется **TCP** (порт 49), что конечно же медленнее, чем UDP у RADIUS.
- 2) Протокол шифрует все данные между AAA-сервером и AAA-клиентом.
- 3) Процессы аутентификации и авторизации абсолютно независимы. Это позволяет выполнять авторизацию каждой команды, которую вводит пользователь. Таким образом мы получаем более гибкий контроль всех действий.
- 4) Проприетарный протокол (т.е. закрытый).

Самый известный TACACS+ сервер - Cisco Secure Access Control Server (**ACS**). Это также проприетарная разработка компании Cisco. Продукт платный, но имеется демо период (возможность бесплатного тестирования).

1.3.3 RADIUS vs TACACS+

Какой же протокол выбрать для своей сети? Для удобства я вынес в табличку основные характеристики этих протоколов:

	RADIUS	TACACS+
Транспортный протокол	UDP 1812/1645 (authentication) 1813/1646 (accounting)	TCP, порт 49
Шифрование	Шифрует только пароль	Шифрует все данные

Особенности	Открытый протокол. “Плохо” реализована авторизация. Детальный учет (accounting).	Проприетарный протокол. Гибкая авторизация команд. Менее детальный учет.
-------------	--	--

Ответ как всегда зависит от каждой конкретной задачи.

Есть мнение, что у RADIUS гораздо лучше выполнен учет (accounting). Однако, лично в моей практике, не было таких случаев, когда данное преимущество играло важную роль (хотя для Интернет провайдеров это очень критично). А вот авторизация каждой команды, это действительно важное отличие и здесь TACACS выигрывает. RADIUS чаще всего используется для аутентификации удаленных пользователей, т.е. VPN подключений, либо для WiFi пользователей. Также RADIUS сервер встречается в сети любого Интернет провайдера, где нужно аутентифицировать сотни и тысячи пользователей. TACACS же чаще всего применяется для аутентификации и авторизации администраторов на сетевых устройствах. Опять же, данный выбор обусловлен более гибкой авторизацией, когда можно проверять буквально каждую команду. При этом RADIUS поддерживается практически любым сетевым оборудованием, чего не скажешь о TACACS.

Данные протоколы весьма часто используются совместно, распределяя задачи. Руководствуясь личным опытом, могу дать следующий совет:

*Для аутентификации VPN-пользователей и WiFi-клиентов используйте **RADIUS**.
 Для аутентификации и авторизации администраторов на сетевом оборудовании -
TACACS+.*

1.3.4 Настройка AAA-сервера

Прежде чем перейти к описанию настроек сетевого оборудования будет весьма полезно рассмотреть настройки AAA-сервера. Хотя данная тема и выходит за рамки нашей книги, я все же опишу основные моменты. В качестве примера мы будем рассматривать протокол TACACS+, поскольку он гораздо интереснее в плане безопасности, т.к. позволяет авторизовать каждую команду.

В качестве AAA-сервера можно выбрать **Cisco ACS** в триальном режиме. Автор использовал проект [Tacacs GUI plus](#), который совершенно бесплатен и предоставляет графический Web-интерфейс. Настройка AAA-сервера состоит из трех этапов:

- 1) **Создание устройства**. Как показано на рисунке ниже, мы добавляем маршрутизатор **R1**, указываем его ip-адрес и ключ (**Tacacs Key**), который используется для шифрования данных между маршрутизатором и AAA-сервером.

Edit Device R1



Device Name

R1

Type

Router

E.g., Router or Switch

IP

192.168.56.254

Tacacs Key

.....



Required for the secure connection setup between server and device

- 2) **Создание уровней.** Здесь подразумеваются уровни привилегий (**Privilege Level**). Мы также задаем уровни при использовании локальной базы (**local**). На рисунке ниже мы создали уровень 15 (**Level15**) и разрешили все команды.

Edit Rule Level15



Name

Level15

Privilege Level



Privilege Level 15

Choose Access Restriction

- Allow All Commands
- Set Allowed Commands
- Set Auto Command

Added 2016-11-27 18:28:59

Last Modified 2016-11-27 18:28:59

Close

Edit Rule

Для примера создадим еще один уровень (**Level1**), которому разрешим всего несколько команд: **ping**, **show run** и **exit**.

Add New Rule



Name

Privilege Level



Privilege Level 1

Choose Access Restriction

- Allow All Commands
- Set Allowed Commands
- Set Auto Command

Allowed Commands

Write in format <command> { permit|deny <expression>}. Use |=| characters between commands.

Close

Add Rule

- 3) **Создание пользователей.** Последний этап. Создаем пользователя, указываем пароль и его уровень привилегий. В нашем примере создан пользователь **admin15** с паролем **admin15** и уровнем привилегий **Level15** (т.е. ему доступны все команды).

User name**Password** Select if you have entered encrypted password**How to store user password?** Encrypt Clear Text**Enable Password (optional)**

Used as a standard Enable Password

 Select if you have entered encrypted password**How to store enable password?** Encrypt Clear Text**Access Rule:**

Choose preconfigured access rule

Точно таким же способ был создан пользователь **admin1** с паролем **admin1** и уровнем **Level1**. Согласно нашим настройкам, ему будут доступны всего три команды.

На этом базовая настройка AAA-сервера завершена. Настройка Cisco ACS имеет весьма схожий процесс и вряд ли может вызвать какие-то затруднения.

1.3.5 Настройка Authentication

Теперь рассмотрим процесс настройки сетевого оборудования для использования функций AAA. Данную тему нельзя назвать простой, поэтому я настоятельно рекомендую потренироваться на лабораторном макете, прежде чем начинать настройку реального оборудования. Может получиться, что вы просто потеряете доступ.

Для начала рассмотрим настройку Authentication (первая А). Это позволит избежать некоторой путаницы в голове. Процесс настройки можно разбить на 4 основных этапа:

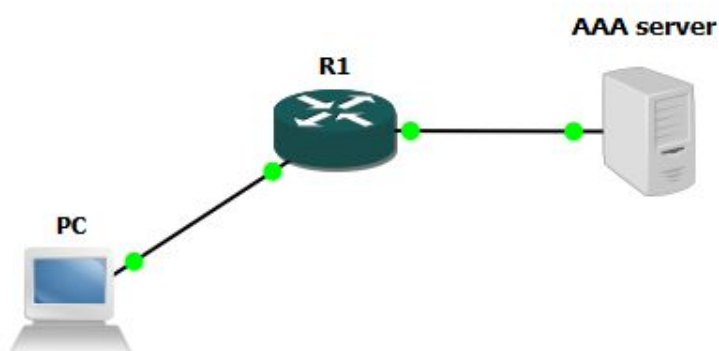
- 1) Создание пользователя в локальной базе. Собственно мы уже делали это ранее (**username admin privilege 15 secret cisco**). Здесь может возникнуть некоторое недопонимание. Мы же используем AAA для того, чтобы централизованно хранить все учетные записи. Так зачем нам локальная учетная запись? А теперь представьте, что по какой-то причине AAA-сервер стал недоступен. Это может случиться из-за сетевого сбоя, либо из-за

хакерской атаки. Тогда вы просто не сможете зайти на оборудование, т.к. сетевое устройство будет пытаться обратиться к AAA-серверу для проверки введенных данных. В этом случае и необходима локальная учетная запись. По сути, она является резервным вариантом входа, когда нет возможности обратиться к AAA-серверу.

Никогда не настраивайте вход ТОЛЬКО через AAA-сервер. Вы не сможете зайти на устройство, если сервер станет недоступен. Используйте локальную учетную запись, как резервный вариант входа.

- 2) Включение функции AAA. Данное действие мы тоже уже выполняли (**aaa new-model**).
- 3) Настройка AAA-сервера. Здесь имеется в виду указание IP - адреса AAA-сервера (или даже нескольких серверов) и секретного ключа (**key**). Эти данные используются устройством для подключения к AAA-серверу и дальнейшему общению. Ключ используется для шифрования всей передаваемой информации.
- 4) Создание метода аутентификации (**method list**). И здесь мы уже попробовали свои силы (**aaa authentication login default local**). Мы создавали **method list** под названием **default**, что означает его применимость сразу ко всем доступным линиям (console, vty). И в конце команды мы указывали способ аутентификации - **local**, т.е. использование локальной базы учетных записей. Далее мы рассмотрим настройку другого способа - **tacacs** либо **radius**.

Теперь приведем пример настройки маршрутизатора, который будет использовать AAA-сервер в качестве основного метода аутентификации и локальную базу, как резервный вариант. Для макетирования использовалась следующая схема:



Ниже представлена настройка маршрутизатора:

<code>enable secret cisco</code>	/задаем пароль на enable
<code>username admin privilege 15 secret cisco</code>	/создаем локального пользователя
<code>aaa new-model</code>	/включаем AAA
<code>tacacs-server host 192.168.56.101</code>	/указываем адрес AAA-сервера
<code>tacacs-server key cisco123</code>	/указываем ключ
<code>aaa authentication login default group tacacs+ local</code>	/создаем method list

Думаю стоит немного пояснить последнюю команду. В качестве имени **method list** мы использовали **default**, т.е. применили его ко всем линиям. Далее, с помощью

параметра **group**, мы определяем несколько возможных вариантов аутентификации. Первым по приоритету идет **tacacs+**, т.е. если для роутера доступен AAA-сервер, то для аутентификации будет использоваться именно этот способ. Если же по каким-то причинам это невозможно, то будет использован второй способ - **local**, т.е. локальная база учетных записей. В данном случае этот способ является резервным и именно для него мы создали локальную учетку **admin**. Кроме того, мы можем использовать в качестве резервного варианта пароль от **enable**. В этом случае последняя команда выглядела бы так:

```
aaa authentication login default group tacacs+ enable
```

Иногда бывает необходимо создать разные политики доступа к оборудованию. Для примера рассмотрим случай с консольным портом. Предположим, что мы уже настроили аутентификацию с помощью AAA и создали **method list** с именем **default**. Но при этом мы хотим, чтобы вход по консоли был без пароля (крайне не рекомендую). Для этого достаточно создать еще один **method list** и “повесить” его на консольный порт:

```
R1(config)#aaa authentication login FREE none
R1(config)#line console 0
R1(config-line)#login authentication FREE
```

В **method list** с именем **FREE** использует параметр **none**, который разрешает вход без пароля.

Настройка **RADIUS** для аутентификации практически идентична и заключается лишь в замене слова “**tacacs**”.

```
radius-server host 192.168.56.2
radius-server key cisco123
aaa authentication login default group radius local
```

Таким образом мы рассмотрели процесс базовой настройки аутентификации с помощью AAA-сервера. При попытке подключения к маршрутизатору получим следующее:

```
User Access Verification
```

```
Username: admin15
Password:
```

```
R1>
```

Как видим, вход с помощью учетной записи **admin15** - успешен. Но стоит обратить внимание на уровень привилегий. Данный пользователь был создан с уровнем **15 (Level15)**, но знак “>” говорит о том, что мы все еще в пользовательском режиме, т.е. уровень **1**. Причина в отсутствии настроек авторизации (вторая А). Именно авторизация позволяет пользователю при входе на устройство автоматически попадать на свой уровень.

1.3.6 Настройка Authorization

Настройка авторизации немного отличается от настройки аутентификации. Ниже представлен пример конфигурации, где мы разберем каждую команду.

```
aaa authorization exec default group tacacs+ local
```

Для начала мы настраиваем авторизацию для входа в соответствующий режим - пользовательский (**user exec mode**) или привилегированный (**privileged exec mode**). Именно после этой команды мы сможем автоматически попадать в нужный режим сразу после входа на устройство. Думаю значение **default group tacacs+ local** вам и так понятно.

```
aaa authorization config-command
aaa authorization commands 1 default group tacacs+ local
aaa authorization commands 15 default group tacacs+ local
```

Здесь мы настраиваем авторизацию каждой вводимой команды, как на уровне 1, так и на уровне 15.

```
aaa authorization console
```

Последним шагом идет настройка авторизации при входе через консольный порт. Несмотря на ранее введенное имя **default**, авторизация на консольном порту не будет проходить без этой команды.

Теперь если попробовать подключиться к маршрутизатору с помощью учетной записи **admin15**, получим следующее:

```
User Access Verification
```

```
Username: admin15
Password:
```

```
R1#
```

Как видим по знаку **"#"** мы попадаем в нужный нам режим (**privileged exec mode**) без необходимости ввода пароля **enable**.

Можно еще раз проверить работу авторизации и войти под учетной записью **admin1**:

```
User Access Verification
```

```
Username: admin1
Password:
```

```
R1>
```

Как и ожидалось, мы попадаем в пользовательский режим. При этом, если вы помните, данный пользователь ограничен всего тремя командами, в соответствии с настройками на AAA-сервере (это **ping**, **show run** и **exit**). Попробовав выполнить другие команды получим следующее:

```
R1>show cdp neighbors
Command authorization failed.
```

Команда не авторизована и не может быть выполнена. И даже если войти в привилегированный режим с помощью **enable**, мы все равно не сможем выполнить неавторизованные команды:

```
R1>en
Password:
R1#conf t
Command authorization failed.
```

1.3.7 Настройка Accounting

Последним этапом будет настройка учета (**accounting**) или простым языком - логирование всех действий пользователей. Делается это элементарно:

```
aaa accounting exec default start-stop group tacacs+
aaa accounting commands 1 default start-stop group tacacs+
aaa accounting commands 15 default start-stop group tacacs+
```

Здесь мы включили логирование всех вводимых команд, как на уровне 1, так и на уровне 15. После этого на AAA-сервер можно увидеть следующую информацию:

Date/Time	Name Device (IP)	User Name	User IP	Privilege Level	Command
2016-11-30 02:20:23	R1 (192.168.56.254)	admin1	console	1	ping 192.168.56.101
2016-11-30 02:19:24	R1 (192.168.56.254)	admin15	console	15	show running-config
2016-11-30 02:19:24	R1 (192.168.56.254)	admin15	console	15	show running-config
2016-11-30 02:19:17	R1 (192.168.56.254)	admin15	console	15	write memory

Мы видим на каком устройстве, в какое время, кем и как были произведены какие-либо действия. Данная информация может существенно помочь при поиске причин “внезапной” недоступности какого-либо устройства после неудачной настройки. Это и есть основное предназначение функции **Accounting**.

1.3.8 Ограничения AAA

Следует отметить, что мы весьма поверхностно рассмотрели возможности AAA. Описанные выше команды, лишь пример минимально необходимых настроек. Более подробное рассмотрение данной технологии выходит за рамки нашей книги. Однако остался еще один момент, который я не могу обойти стороной.

Ранее было сказано, что при настройке AAA необходимо обязательно оставлять резервный вариант входа в виде локальной учетной записи (либо пароля на enable). Но как же централизованно администрировать эти учетки? Ведь AAA не позволят менять локальные пароли на устройствах. Неужели придется это делать вручную? А если устройств сотни, как у интернет провайдеров? Не менять пароли локальных учеток также плохая идея.

Как правило, для решения данной проблемы используются менеджеры конфигураций (**network configuration manager**) либо самописные скрипты. Яркими примерами менеджеров конфигураций являются **SolarWinds Network Configuration Manager** (платный), и **rConfig** (бесплатный).

*Для централизованной смены паролей локальных учетных записей используйте менеджеры конфигураций (**network configuration manager**).*

Данный софт позволяет централизованно изменять настройки сетевых устройств, даже если они разных производителей (Cisco, Juniper, HP и т.д.). Описание данных программ выходит за рамки нашей книги.

1.4 Парольная политика

Мы уже частично затронули эту тему в предыдущих главах, теперь мы рассмотрим данный вопрос более детально. В современном мире, практически любой человек понимает важность пароля, а точнее важность его секретности. Используете ли вы соц. сети, электронную почту или онлайн банкинг, везде необходим секретный пароль. Тоже самое касается и корпоративных сетей. Пароль должен исключать возможность несанкционированного доступа к сетевому оборудованию. Это касается как доступа из внешней сети (Интернет), так и из внутренней.

Ранее мы уже обсудили, что пароли в конфигурации нужно шифровать, а сами пароли не должны быть элементарными (вроде **cisco** или **password**). Однако и этого недостаточно. Многие пользователи довольно часто используют корпоративные пароли для личных целей, например при регистрации в соц. сетях, torrent - трекерах, различных развлекательных порталах. Признаюсь, я и сам делал подобные вещи несколько раз. Уверен, что каждый слышал о периодических взломах популярных сайтов, в результате чего в сеть “утекали” огромные базы учетных записей. Среди этих записей может быть и ваш корпоративный пароль. И неважно насколько он сложный, т.к. он полностью скомпрометирован. Необходима смена пароля.

Пароли необходимо периодически менять вне зависимости от их сложности.

В данном случае, единственный способ обеспечить должный уровень безопасности это соблюдать парольную политику.

1.4.1 Основные положения

Для начала рассмотрим парольную политику в ее классическом виде и опишем основные моменты. Как правило, парольная политика содержит следующие требования:

- 1) Минимальная длина пароля. На данный момент рекомендуемая длина - не менее 8 символов. Это ограничение связано с временем, которое необходимо для подбора пароля при “брутфорсе” (атака по словарю). Думаю не стоит объяснять, что чем длиннее пароль, тем дольше придется его разгадывать.
- 2) Используемые символы в пароле. Практически все стандарты требуют наличия в пароле букв верхнего и нижнего регистра, цифр и специальных символов (такие знаки как !, @, -, _ и т.д.). Это существенно усложняет процесс атаки по словарю, т.к. возможных комбинаций становится гораздо больше.
- 3) Запрет на распространение паролей. Личный пароль не может быть сообщен кому-либо или записан на бумагу. Многие пользователи любят записывать пароли на стикеры и приклеивать к мониторам. Это недопустимо. В таком случае можно считать, что пароль скомпрометирован.
- 4) Плановая смена паролей. Здесь уже каждая компания сама выбирает временные интервалы. Как правило это либо раз в полгода, либо раз в квартал.
- 5) Внеплановая смена паролей. Обычно производится при увольнении сотрудника, либо при обнаружении факта компрометирования пароля. Ни в

коем случае нельзя оставлять учетные записи пользователей, которые уже не работают в организации.

Большинство организаций ограничиваются данными пунктами. Однако этого недостаточно. Есть один фактор, который мешает соблюдению описанных пунктов - это память человека. Чем сложнее пароль, тем труднее его запомнить. Люди могут идти на различные ухищрения, чтобы выбрать легкий для запоминания пароль, но при этом формально соблюдая парольную политику. Приведем пример. Возьмем первые два пункта: пароль должен содержать минимум 8 символов, включать буквы верхнего/нижнего регистра, цифры и специальные символы. Данные требования легко соблюсти выбрав следующие пароли: **Password1!**, **P@ssw0rd**, **Pa\$\$w0rd** и т.д. Формально парольная политика применена, но данные пароли подбираются за считанные минуты. В сети Интернет есть целые словари с подобными и самыми распространенными паролями. Более того, если будет выполняться "брутфорс", то в первую очередь будут подбираться именно такие, типовые варианты.

Приведем еще один пример, касающийся плановой смены паролей. Предположим, что подошло время смены. Пользователь, дабы не усложнять себе жизнь, меняет старый пароль **Password1!** на новый - **Password2!**. Опять же, формально все соблюдено, но эффективность этой процедуры весьма сомнительна.

Еще один момент. Пункт 3) гласит, что пароль нельзя записывать ни на физический, ни на электронный носитель. Но что делать если пароль забыт, либо что-то случилось с тем, кто его знает? А если это пароль от коммутатора ядра в крупном интернет провайдере? AAA-сервера частично решают эту проблему, но как быть с локальными паролями на устройствах? Без них все равно не обойтись.

Описанные выше проблемы дали повод к появлению еще нескольких пунктов в парольной политике:

6. Пароль не должен содержать легко предугадываемые сочетания символов. Пример: pass, adm, qwert, asdf, 1234, user, либо фразы имеющие отношение к имени/фамилии пользователя.
7. При смене пароля, новый пароль должен отличаться от старого не менее чем на 4 символа.
8. Пароли от особо важных узлов должны быть записаны на какой либо носитель (лист бумаги, флешка) и сохранены в безопасном месте (сейф).

Соблюдайте парольную политику! Если ее нет - разработайте.

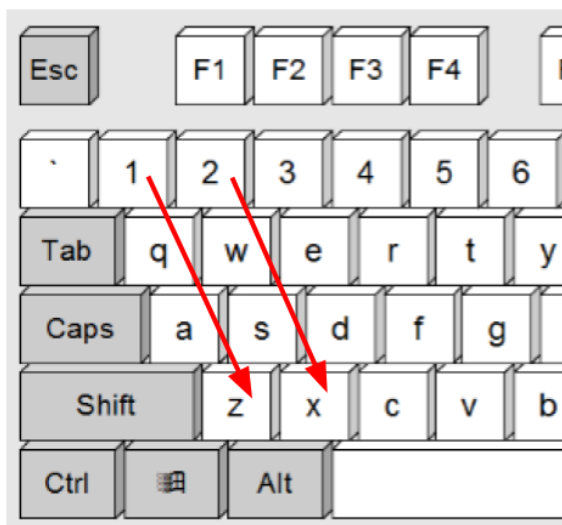
1.4.2 Самые популярные пароли 2015 года

Как было сказано ранее, в сети Интернет можно найти целые словари наиболее часто используемых паролей. Данная база обновляется каждый год и публикуется так называемый ТОП. Делается это с целью показать, насколько часто и насколько "плохие" пароли мы используем. Ниже представлены 25 самых популярных паролей 2015 года:

1. 123456
2. password
3. 12345678
4. qwerty

5. 12345
6. 123456789
7. football
8. 1234
9. 1234567
10. baseball
11. welcome
12. 1234567890
13. abc123
14. 111111
15. 1qaz2wsx
16. dragon
17. master
18. monkey
19. letmein
20. login
21. princess
22. qwertyuiop
23. solo
24. passw0rd
25. starwars

Кроме пресловутых последовательностей цифр и популярных слов, можно заметить пароль **1qaz2wsx**. На первый взгляд он не кажется простым. Но это только на первый взгляд. Очень большое количество пользователей любят придумывать пароли, которые удобно вводить с клавиатуры, ведь так легче запомнить:



Можно привести много подобных примеров: **qaz123**, **qweasd**, **zaq1xsw2**, **qazxsw** и т.д. При этом многие администраторы считают, что они придумали хороший пароль. Это в корне не верно. Современные алгоритмы подбора паролей учитывают данные “привычки” и позволяют угадывать такие комбинации за считанные секунды.

Не используйте клавиатурные комбинации при создании пароля.

1.4.3 Пароли по умолчанию

Очень многие администраторы забывают о такой важной вещи, как “Пароль по умолчанию”. IP - камера, ip - телефон, источник бесперебойного питания, сетевой принтер или даже холодильник (который подключается к сети Интернет) имеют дефолтные пароли. Большинство беспечно полагают, что эти устройства не представляют какой-либо опасности. Это серьезное заблуждение. Пароли для таких устройств можно с легкостью найти в сети Интернет, что позволяет злоумышленнику весьма просто подключиться к устройству, которое затем может использоваться как плацдарм для дальнейшей атаки. Ситуация ухудшается с увеличением сегмента Интернета вещей, так называемый **IoT** (Internet of Things). Яркий тому пример - **ботнет Mirai**.

Mirai - одна из самых больших ботнет сетей на сегодняшний день. Однако больше всего поражает не размер сети, а то, как эта сеть появилась. Метод гениально прост. Ботнет не использует какую-то сложную уязвимость софта. Все, что он делает - подбирает пароль. Тот самый “пароль по умолчанию”, который устанавливают производители различных устройств. Mirai использует всего 61 различную комбинацию логин/пароль методом перебора (возможно сейчас этих комбинаций уже больше). Столь малое количество комбинаций обеспечивает высокую скорость обнаружения уязвимых устройств, которые в дальнейшем используются в ботнет сети. На момент написания статьи ботнет насчитывал уже более 1 миллиона устройств IoT. На что способен Mirai? С помощью Mirai была организована самая мощная, в истории Интернет, DDoS атака - 665 Гбит/с. Эксперты утверждают, что мощность атак будет только расти. А виной всему дефолтный пароль, который забыли сменить при установке устройства.

Ни в коем случае не оставляйте пароли по умолчанию.

1.4.4 Фишинг

Последнее, о чем я бы хотел поговорить в этой главе - **Фишинг** (от английского fishing - рыбалка). Название говорит само за себя. Злоумышленник кидает крючок с приманкой и ждет пока клюнет жертва. В качестве приманки может выступать поддельный сайт социальной сети, куда хакер перенаправляет пользователя, где последний вводит логин/пароль, не подозревая о зловредности сайта. Злоумышленник получает нужные данные и использует их в своих целях. Как говорилось выше, некоторые пользователи зачастую используют корпоративные пароли для сторонних сервисов, что существенно повышает риск для сети компании.

Еще один популярный инструмент для фишинга - email рассылка. Пользователю отправляется поддельное письмо (например от банка или начальника) с просьбой передать пароль или любые другие ценные данные (номер кредитной карты). Данный тип атак зачастую совмещает элементы социальной инженерии, что существенно повышает их эффективность.

К сожалению, данный тип атак невозможно предотвратить средствами, которые мы рассматриваем в рамках этой книги (коммутаторы, маршрутизаторы). Единственный выход - информирование ваших пользователей/администраторов о

возможных опасностях, либо использование специализированных средств защиты от фишинга (о которых мы обязательно поговорим, но уже в следующей книге).

1.5 Чек-лист №1

Первая глава книги подошла к концу и хотелось бы резюмировать вышесказанное. Для этого запишем ключевые мысли предыдущих параграфов в виде чек-листа:

- 1) *Ограничьте физический доступ к вашему сетевому оборудованию, а также кабельной системе. Это снизит риски как случайных, так и целенаправленных вторжений.*
- 2) *Закрывайте доступ к консольному порту даже для пользовательского режима.*
- 3) *Не используйте **enable password** даже в связке с **service password-encryption**. Используйте сложные пароли и **enable secret**.*
- 4) *НИКОГДА не используйте стандартные учетные записи. Придумывайте уникальные имена пользователей.*
- 5) *Не создавайте учетные записи с уровнем привилегий 15. Пароль учетной записи и пароль на **enable** должны быть разными!*
- 6) *Для настройки аутентификации используйте **aaa new-model**.*
- 7) *Запретите сброс пароля (**no service password-recovery**) на коммутаторах уровня доступа к которым возможен физический доступ.*
- 8) *Для настройки удаленного доступа используйте **aaa new-model**.*
- 9) *Не используйте **Telnet** в качестве протокола удаленного подключения.*
- 10) *Используйте защищенные протоколы удаленного подключения (например **SSHv2**).*
- 11) *При необходимости организации web-доступа к оборудованию, используйте **HTTPS** вместо **HTTP**. Помните об огромном количестве уязвимостей, связанных с **HTTP/S**.*
- 12) *Ограничьте удаленный доступ. Лишь несколько компьютеров должны иметь возможность подключения к оборудованию.*
- 13) *Для защиты от подбора паролей (**Brute Force**) настройте временную блокировку повторной аутентификации.*
- 14) *Задавайте имя устройства (**hostname**) и сообщение после аутентификации (**banner exec**), чтобы в будущем не “перепутать” оборудование.*
- 15) *Если в вашей сети более 10-и устройств, то вы просто обязаны использовать AAA-сервер.*
- 16) *Для аутентификации VPN-пользователей и WiFi-клиентов используйте **RADIUS**. Для аутентификации и авторизации администраторов на сетевом оборудовании - **TACACS+**.*
- 17) *Никогда не настраивайте вход ТОЛЬКО через AAA-сервер. Вы не сможете зайти на устройство, если сервер станет недоступен. Используйте локальную учетную запись, как резервный вариант входа.*
- 18) *Для централизованной смены паролей локальных учетных записей используйте менеджеры конфигураций (**network configuration manager**).*
- 19) *Пароли необходимо периодически менять вне зависимости от их сложности.*

20) Соблюдайте парольную политику! Если ее нет - разработайте.

21) Не используйте клавиатурные комбинации при создании пароля.

22) Ни в коем случае не оставляйте пароли по умолчанию.

Придерживаясь этих советов, при организации доступа к оборудованию, вы существенно повысите безопасность вашей сети. При этом от вас не требуются какие-либо глубокие познания в области информационной безопасности.

1.6 Пример конфигурации

Для закрепления материала приведем обобщающий пример настройки доступа для роутера с использованием AAA-сервера:

1) Задаем пароль на **enable** и создаем пользователя

```
Router#configure terminal
Router(config)#enable secret cisco
Router(config)#username admin privilege 1 secret cisco
Router(config)#service password-encryption
Router(config)#no service password-recovery
```

2) Задаем **hostname** и банер

```
Router(config)# hostname R1
R1(config)#banner exec c
Enter TEXT message. End with the character 'c'.
Test banner for NetSkills
c
R1(config)#
```

3) Настраиваем **SSH**

```
Router(config)#ip domain-name netskills.ru
Router(config)#crypto key generate rsa modulus 1024
Router(config)#ip ssh version 2
Router(config)#ip ssh time-out 15
Router(config)#ip ssh logging events
Router(config)#line vty 0 4
Router(config-line)#transport input ssh
Router(config-line)#exec-timeout 5 0
Router(config-line)#exit
```

4) Отключаем **HTTP** и **HTTPS**

```
Router(config)#no ip http secure-server
Router(config)#no ip http server
```

5) Ограничиваем доступ для определенных ip-адресов

```
Router(config)#ip access-list standard SSH-ACCESS
Router(config-std-nacl)#permit host 192.168.2.2
Router(config-std-nacl)#permit host 192.168.2.3
Router(config-std-nacl)#exit
Router(config)#line vty 0 4
Router(config-line)#access-class SSH-ACCESS in
```

6) Настраиваем защиту от **brute force**

```
Router(config)#login delay 5
Router(config)#login block-for 60 attempts 3 within 30
```

7) Настраиваем **AAA**

```
R1(config)#aaa new-model
R1(config)#tacacs-server host 192.168.56.101
R1(config)#tacacs-server key cisco123
R1(config)#aaa authentication login default group tacacs+ local
R1(config)#aaa authorization exec default group tacacs+ local
R1(config)#aaa authorization config-command
R1(config)#aaa authorization commands 1 default group tacacs+ local
R1(config)#aaa authorization commands 15 default group tacacs+ local
R1(config)#aaa authorization console
R1(config)#aaa accounting exec default start-stop group tacacs+
R1(config)#aaa accounting commands 1 default start-stop group tacacs+
R1(config)#aaa accounting commands 15 default start-stop group tacacs+
```

Как было сказано выше, это всего лишь пример и ваши настройки могут конечно же отличаться.

2. Лучшие практики

Перед началом “серьезной” части книги, мне бы хотелось рассмотреть еще несколько аспектов сетевой безопасности. Советы из данной главы относятся к категории “**best practice**” и их можно с легкостью применить к любой сети без каких-то глобальных “переделок”.

2.1 Logs

Если вы когда-нибудь посещали уроки Истории, то наверняка слышали фразу вроде: “Будущее невозможно без знаний о прошлом”. Это очень глубокая мысль, которая имеет применение и в сфере информационной безопасности. В нашем случае в качестве учебника по истории будут выступать **Логи (logs)**. Их также называют журналом событий, файлом регистраций и т.д. Название не столь важно, а смысл один - записи о всех событиях в хронологическом порядке. Что конкретно подразумевается под “событиями” мы обсудим чуть позже.

Меня всегда удивляло, с какой халатностью относятся к логам многие системные администраторы. Их (логи) либо не собирают, либо просто не обращают на них внимания. А между тем, логи, это один из самых мощных инструментов как при поиске неисправностей (**troubleshooting**), так и при расследовании различных инцидентов (в том числе ИБ). Сколько было неудачных попыток доступа к маршрутизатору? Когда отключился порт fa0/1 и происходило ли это раньше? Когда произошло падение VPN-туннеля? Сколько прошло времени с перезагрузки устройства и связано ли это с отключением электричества? Когда были внесены последние изменения в конфигурацию? На эти и многие другие вопросы мы можем ответить только обладая логами.

Информационная безопасность невозможна без использования логов.

Логирование (**logging**) позволяет видеть практически все, что происходило в вашей сети. Согласитесь, что невозможно наблюдать за всем оборудованием в режиме реального времени, особенно если инциденты происходят в ночное время, а сеть исчисляется десятками коммутаторов и маршрутизаторов. Кроме того, периодический просмотр логов позволяет избежать проблем, которые могут случиться в будущем.

К сожалению сетевое оборудование Cisco (и других вендоров) имеет весьма ограниченное место для логов (буфер), что в свою очередь сказывается на временном промежутке, который может быть отражен в логах. При исчерпании размера буфера, самые старые логи просто удаляются. Кроме того, при перезагрузке устройства логи будут потеряны безвозвратно. Для решения данных проблем используются **Лог-серверы**, но обо всем по порядку.

2.1.1 Методы сбора логов

Всего существует 6 способов сбора логов с оборудования Cisco и не только. Рассмотрим их:

- 1) **Console Logging** - Вывод сообщений в консоль. Данный способ работает по умолчанию и выводит логи прямо в консоль устройства.
- 2) **Buffered Logging** - Сохранение логов в буфер устройства, т.е. RAM память.
- 3) **Terminal Logging** - Вывод логов в терминал, т.е. для Telnet или SSH сессий.
- 4) **Syslog сервер** - централизованный сбор логов по протоколу Syslog.
- 5) **SNMP Traps** - централизованный сбор логов по протоколу SNMP.
- 6) **AAA** - использование Accounting. Сбор логов касательно подключения к оборудованию и ввода команд. Мы уже рассматривали данный метод.

Каждый из способов обладает своими достоинствами и недостатками, поэтому необходимо использовать их вместе. Ниже мы рассмотрим некоторые аспекты этих методов.

Используйте несколько способов сбора логов.

2.1.2 Уровни логирования

Чуть выше мы сформулировали, что Логи это записи о всех событиях в хронологическом порядке. Но что является событием? Все зависит от уровня логирования. Именно он определяет, какую информацию необходимо отображать в логах. Всего существует 8 уровней логирования:

- 0 - Emergencies.** События связанные с неработоспособностью системы.
- 1 - Alerts.** Сообщения о необходимости немедленного вмешательства.
- 2 - Critical.** Критические события.
- 3 - Errors.** Сообщения об ошибках.
- 4 - Warnings.** Сообщения содержащие предупреждения.
- 5 - Notifications.** Важные уведомления.
- 6 - Informational.** Информационные сообщения.
- 7 - Debugging.** Отладочные сообщения.

Данные уровни обладают наследственностью, т.е. выбрав уровень 7, вы будете получать сообщения всех уровней (от 0 до 7), а выбрав уровень 3 - только от 0 до 3. С осторожностью повышайте уровень логирования, поскольку чем выше уровень, тем больше нагрузка на CPU. К примеру, если через маршрутизатор идет "большой" трафик и процессор уже находится под высокой нагрузкой, то включив 7 уровень (Debugging) вы можете просто потерять управление над устройством.

С осторожностью используйте повышенные уровни логирования.

В корпоративных сетях чаще всего применяется 6-ой (Informational) уровень логирования. Уровень 7 (Debugging) обычно используется при поиске неисправностей (troubleshooting), например для определения проблем с построением VPN туннеля.

2.1.3 Console Logging

Подключившись к коммутатору или маршрутизатору по консоли вы сразу увидите отображение логов. По умолчанию включен 5-ый уровень логирования. Если устройство находится под нагрузкой (т.е. через него идет трафик), то ни в коем случае не стоит повышать уровень логирования, иначе вы просто не сможете работать из консоли, т.к. постоянно появляющиеся логи не дадут вам набрать команды. Кроме того, если вы просто вытащите консольный кабель, то логи будут продолжать отсылаться в консоль и отнимать существенные ресурсы устройства.

*Используйте **Console Logging** только при необходимости.*

Однако бывают случаи (обычно на этапе тестирования), когда Console Logging наиболее удобен, т.к. позволяет наблюдать за устройством в реальном времени. Для настройки уровня логирования используйте команду:

```
Router(config)#logging console 7
```

или

```
Router(config)#logging console debugging
```

Если логи начинают мешать работе в консоли, то либо измените уровень логирования, либо просто отключите их командой:

```
R1(config)#no logging console
```

2.1.4 Buffered Logging

Как было сказано выше, логи можно хранить на самом устройстве, в так называемом буфере - RAM память. При этом вы можете изменять размер этого буфера, тем самым регулируя "глубину" логирования. Следует весьма аккуратно подходить к выбору размера буфера. Как правило он зависит от свободной оперативной памяти устройства. Настроив слишком маленький буфер вы рискуете упустить важные события, которые будут затерты при исчерпании свободного места. Выбрав слишком большой буфер вы можете занять всю свободную оперативную память устройства, что приведет к неизвестным последствиям (обычно это зависание).

Как правило размер буфера устанавливается в 16, либо 32 Кбайта. Более современные устройства позволяют выделять под логи несколько мегабайт оперативной памяти. Настройка элементарна:

```
Router(config)# logging on  
Router(config)# logging buffered 32768  
Router(config)# logging buffered informational
```

В данном случае мы сначала включили Buffered Logging, затем установили размер буфера в 32 Кбайта и выбрали уровень логирования informational (т.е. 6-ой). Для просмотра логов используйте команду **show log** из привилегированного режима.

*Настройте **Buffered Logging** даже при использовании **Syslog-сервера**.*

Ни в коем случае не пренебрегайте данным методом сбора логов, даже если вы используете Syslog-сервер. Может быть ситуация, когда Syslog-сервер окажется недоступен и тогда логи будут безвозвратно потеряны, если вы не используете Buffered Logging.

2.1.5 Terminal Logging

Как было сказано ранее, иногда бывают ситуации, когда нужно видеть логи в режиме реального времени, как в примере с Console Logging. В этом случае вывод логов прямо в консоль устройства намного удобнее, чем постоянно набирать команду **show log** для отображения сообщений из буфера устройства. Но не всегда есть возможность использовать консоль. Как быть при удаленном подключении (Telnet, SSH)? Для решения этой проблемы существует команда **terminal monitor**, которая и позволяет выводить логи в терминальную сессию. При этом вы также можете задать уровень логирования:

```
Router(config)#logging monitor informational
Router(config)#exit
Router#terminal monitor
```

Обратите внимание, что команда **terminal monitor** вводится из привилегированного режима, а не из режима глобальной конфигурации. Для отключения логов воспользуйтесь командой **terminal no monitor**.

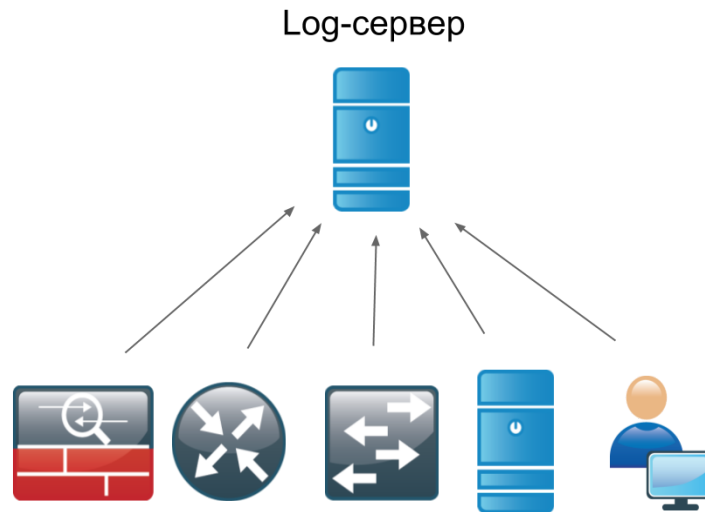
Включая **terminal monitor** помните об увеличении нагрузки на устройство и важности уровней логирования. Также стоит учитывать пропускную способность канала по которому вы подключены. При большом объеме логов канал может оказаться полностью забит, что приведет к разрыву терминальной сессии (т.е. вы потеряете управление).

*Используйте **Terminal Logging** с осторожностью и только в случае острой необходимости.*

От себя могу добавить, что использовать Terminal Logging необходимо в весьма редких ситуациях. В остальном, гораздо удобнее использовать Syslog-сервер.

2.1.6 Syslog-сервер

Главная задача **Лог-сервера** - централизованный сбор логов со всех сетевых устройств.



Логи собираются на выделенном Лог-сервере, который может иметь огромное дисковое пространство, что позволяет хранить события за гораздо больший временной интервал (6, 12 месяцев и даже больше). Мониторинг существенно упрощается при использовании Лог-сервера, т.к. в этом случае нет нужды подключаться к каждому устройству для проверки журнала событий. Лог-сервер является обязательным элементом любой более-менее серьезной сети.

Сбор логов осуществляется с помощью специализированных протоколов. **Syslog** является практически стандартом, поэтому Syslog-сервер часто называют просто Лог-сервер. Данный протокол использует **514 (UDP)** порт, а все данные пересылаются в открытом виде. Еще одним популярным методом сбора Логов является **SNMP Traps**. Выбирая Лог-сервер, убедитесь, что он поддерживает данные технологии.

В качестве примера Лог-сервера можно привести [Kiwi Syslog сервер](#), который имеет бесплатную версию для Windows систем. Для Linux систем выбор гораздо больше. Есть множество вариантов, как бесплатных, так и платных, стоимость которых достигает сотни тысяч долларов.

Используйте Лог-серверы для централизованного сбора логов.

Здесь стоит отметить, что хранение Логов не является панацеей. Представьте, что ежедневно в вашей сети генерируется более миллиона событий (а это весьма скромный показатель для сети средних размеров). При этом в Логах содержится не только полезная информация. Как разобраться в этом хаосе? Именно поэтому сейчас становятся очень популярными Лог-серверы, которые обладают функцией расширенного поиска и автоматическим анализом и корреляцией событий. Это позволяет выделять самое главное из огромной массы событий. Такие Лог-серверы относятся к категории **Log Management** или **SIEM** системам. Данная тема очень обширна и заслуживает отдельной книги, поэтому мы не будем рассматривать ее более подробно. Однако, мне хотелось бы отметить несколько продуктов, которые заслуживают внимания (по субъективному мнению автора):

- 1) [Splunk](#)
- 2) [Graylog](#)

3) [OSSIM](#)

4) [ELK \(Elastic Stack\)](#)

Дабы не перегружать книгу сторонней информацией, оставлю данную тему для самостоятельного ознакомления. Могу лишь добавить, что если вы пользуетесь Логами в своей ежедневной работе, то рано или поздно вам придется отказаться от классических Лог-серверов в пользу систем с функцией анализа и корреляции событий. Это своего рода эволюция в сегменте продуктов обработки данных.

Теперь рассмотрим элементарный процесс настройки оборудования Cisco для отправки Логов на выделенный Лог-сервер:

```
Router(config)#logging host 192.168.1.100   \адрес Syslog-сервера
Router(config)#logging trap informational   \6-ой уровень логирования
```

Кроме того, вы можете настроить категории логов (**facilities**), на основе которых Syslog-сервер будет сортировать все сообщения. Однако, данная настройка сильно зависит от выбранного сервера, поэтому мы не будем раскрывать данную тему.

2.1.7 SNMP Traps

Это последний метод сбора логов, который мы рассмотрим (т.к. AAA мы рассмотрели ранее). Самое главное преимущество **SNMP Traps** - возможность передачи логов касающихся конкретного параметра устройства, например: температура процессора, напряжение сети, изменение в конфигурации и т.д. Это существенно отличается от “классических” уровней логирования, где логи просто разбиты на категории. С помощью SNMP Traps можно существенно сократить количество логов и мониторить только действительно нужные параметры, исключая остальной “мусор”. Тема SNMP весьма обширна и требует изложения серьезной теоретической базы, поэтому, с вашего позволения, мы не будем рассматривать эту тему в рамках данной книги.

2.1.8 Безопасность логов

Возвращаясь к теме Syslog-сервера стоит помнить, что все логи передаются в совершенно открытом виде. Весьма часто в логах содержится очень важная информация, которой может воспользоваться злоумышленник. Чтобы защитить Syslog-сообщения от несанкционированного доступа лучше всего использовать выделенный сегмент (VLAN) для всего управляющего трафика, как это описано в пункте 1.2.5. Кроме того, сам Лог-сервер представляет большой интерес для хакера. Злоумышленник может либо получить нужную информацию, либо удалить сообщения, связанные со взломом системы, что позволит ему “замести следы” своей атаки. Например он может удалить все сообщения связанные с неудачным вводом пароля, которые были сгенерированы при brute force атаке.

Еще одним примером атаки на Лог-сервер может быть простое переполнение, когда злоумышленник генерирует огромное количество ложных сообщений. В результате на Лог-сервере просто кончается место для обработки настоящих событий. Это открывает для хакера возможность быть незамеченным при взломе системы. Учитывая описанные риски, заранее позаботьтесь о защите Лог-сервера.

Первое, что нужно сделать, это ограничить доступ к Syslog-серверу с помощью access-list-ов. Причем не только доступ к администрированию сервера (SSH, HTTPS и т.д.), но и доступ устройств к серверу по протоколу syslog (UDP, порт 514). Т.е. отправлять логи на сервер могут только те устройства, которым вы открыли доступ.

Некоторые события (DDoS) также генерируют огромное количество лог сообщений, что в итоге может перегрузить Лог-сервер. Чтобы этого избежать в Cisco IOS предусмотрена возможность ограничения пересылаемых логов за единицу времени. Ограничение настраивается следующим образом:

```
Router(config)#logging rate-limit all 50
```

В данном случае мы ограничили отправку syslog до 50 сообщений в секунду.

*Уделите особое внимание защите **Syslog-сервера**.*

2.1.9 Время

Логи совершенно бесполезны если по ним нельзя точно определить, когда именно произошло то или иное событие. Именно поэтому нужно убедиться, что на всех сетевых устройствах установлено корректное время. Настройку времени можно производить вручную, но в таком случае вы не сможете с точностью до секунды синхронизировать время на всем оборудовании. Расхождение даже в пару секунд может существенно усложнить анализ логов, особенно при расследовании ИБ инцидентов. Для решения этой проблемы используется протокол **NTP**.

Network Time Protocol (NTP) - протокол сетевого времени, если дословно. NTP используется для синхронизации времени. Мы не будем подробно останавливаться на описании этого протокола. Идея заключается лишь в том, что логи должны сопровождаться актуальным временем, а это обязывает нас использовать **NTP-сервер**. Настройка оборудования Cisco как NTP-клиента:

```
Router(config)#ntp server 192.168.1.100      \указывает NTP-сервер
```

Кроме того, необходимо настроить часовой пояс (в моем случае время по Москве):

```
Router(config)#clock timezone MSK 4\часовой пояс +4
```

Но этого недостаточно. Перед отправкой логов, они должны быть снабжены специальными временными метками, иначе мы не сможем определить время по сообщению на Лог-сервере. Для установки временных меток используется команда:

```
Router(config)#service timestamps log datetime msec localtime show-timezone
```

В результате, каждое лог-сообщение будет снабжаться временной меткой из которой можно узнать время (с точностью до миллисекунды) и часовой пояс.

*Не забывайте про настройку **NTP** и **Timestamps**.*

2.2 Лишние сервисы

В первой главе книги (пункт 1.1.1) я упомянул такой протокол как **CDP**, который по умолчанию включен почти на всех устройствах Cisco (речь идет о коммутаторах и маршрутизаторах). CDP позволяет обнаруживать соседние устройства, включая информацию о прошивке, платформе, ip-адресе и т.д. За всю свою многолетнюю практику я лишь пару раз видел, чтобы CDP действительно использовался системными администраторами по назначению. В 99% случаев, данный протокол просто продолжал работать в фоновом режиме. Но самое интересное, что таких “лишних” (неиспользуемых) протоколов или сервисов довольно много (например **HTTP** или **Finger**). И почти все имеют свои потенциальные уязвимости, что оставляет для хакера возможные лазейки.

Ниже приведены команды, которые позволяют отключить те сервисы, которые в большинстве случаев просто не используются:

```
no service tcp-small-servers echo
no service tcp-small-servers discard
no service tcp-small-servers daytime
no service tcp-small-servers chargen
no service udp-small-servers echo
no service udp-small-servers discard
no service udp-small-servers daytime
no service udp-small-servers chargen
no ip finger
no ip bootp server
no ip dhcp boot ignore
no service dhcp
no mop enabled
no ip domain-lookup
no service pad
no ip http server
no ip http secure-server
no service config
no cdp enable
no cdp run
no lldp transmit
no lldp receive
no lldp run global
```

Большинство этих команд выполняется из режима глобальной конфигурации (т.е. после **conf t**). Наличие некоторых команд сильно зависит от версии прошивки. Мы не будем описывать каждый из этих сервисов в целях экономии времени читателя.

Неиспользуемые сервисы весьма опасны, т.к. обычно администратор не занимается их настройкой, а значит все параметры имеют значения по умолчанию, что в итоге позволяет предсказать поведение системы во время специализированных хакерских атак. Отключение неиспользуемых сервисов существенно повысит безопасность вашей сети.

Отключите неиспользуемые сервисы.

2.3 Резервная память

Многие хакерские атаки нацелены на исчерпание ресурсов узла, т.е. **DoS** или **DDoS**. Делается это либо с целью создания обычного простоя, либо для эксплуатации некоторой уязвимости. Для того, чтобы правильно отреагировать на ту или иную атаку, нужно для начала понять ее природу. Это возможно только при подключении к устройству. Однако, при таких атаках почти всегда невозможен ни удаленный, ни локальный (через консоль) доступ к оборудованию. У устройства просто заканчивается свободная оперативная память, которая необходима для создания консольной или терминальной сессии. Такая проблема может возникнуть не только при хакерской атаке, но и при неверной настройке, которая привела к исчерпанию ресурсов устройства.

В относительно новых прошивках Cisco появилась возможность резервирования RAM памяти для консольного подключения. Это значит, что даже при высокоинтенсивной атаке на ваш маршрутизатор, сохраняется достаточное количество ресурсов для консольной сессии. Это позволит подключиться к устройству и увидеть, что именно происходит. Настройка элементарна:

```
Router(config)#memory reserve console 4096          \резервируем 4 Мб
```

В данном случае мы зарезервировали 4 Мб оперативной памяти. Обычно этого хватает для консольной сессии.

Зарезервируйте оперативную память для консольного подключения.

2.4 Защищенные протоколы

Ранее мы уже описали, что весьма опасно использовать **Telnet**, т.к. все данные передаются в открытом виде и злоумышленник может их с легкостью перехватить. В качестве альтернативы был предложен протокол **SSH**, который шифрует все по умолчанию. Аналогичная ситуация с **HTTP** и **HTTPS**.

Однако, при работе с сетевым оборудованием редко удается обойтись вышеуказанными протоколами. Ниже мы рассмотрим еще несколько популярных протоколов, а точнее их более защищенные версии.

2.4.1 SCP

Перед системными администраторами довольно часто возникает задача по обновлению прошивки оборудования. Для этого необходимо “закинуть” новую прошивку на устройство, что обычно делается с помощью **TFTP** или **FTP**-сервера. Технология стара как мир, да и подобный сервер разворачивается в несколько кликов.

Данные протоколы (TFTP и FTP) также нередко используются для резервного копирования конфигураций устройств. Наверняка большинству знакома команда вроде:

```
Router#copy running-config tftp:
```

Этой командой мы копируем текущую конфигурацию на удаленный TFTP-сервер.

Либо обратная ситуация, когда TFTP или FTP-сервер используется для восстановления конфигурации:

```
Router#copy tftp: running-config
```

И если в случае с прошивками нет ничего криминального, то при работе с конфигурациями совершенно недопустимо использовать незащищенный TFTP или FTP-сервер. По аналогии с Telnet, эти протоколы передают все данные в открытом виде, что позволяет злоумышленнику перехватить весьма ценную информацию - конфигурации ваших устройств. Для решения данной проблемы существует более защищенный протокол, такой как **SCP**.

SCP (secure copy) - протокол копирования файлов, который в качестве транспорта использует SSH (т.е. все передаваемые файлы шифруются). Для работы необходим **SCP-сервер**, которым может являться любой Linux дистрибутив с включенным SSH-сервером. Для Windows имеется специализированное программное обеспечение, например [Solarwinds SFTP/SCP Server](#) (доступна бесплатная версия). Сам процесс резервного копирования выглядит следующим образом:

```
Router#copy running-config scp://user:password@192.168.1.100/Cisco-Conf/Router1.config
```

Данной командой мы копируем текущую конфигурацию на SCP-сервер с ip-адресом **192.168.1.100** в папку **Cisco-Conf**, а сам файл будет называться **Router1.config**. Для подключения к SCP-серверу используется логин и пароль, которые должны быть предварительно созданы на сервере. При этом вся передаваемая информация шифруется.

*Используйте **SCP** вместо **TFTP**.*

2.4.2 SNMP

SNMP - Simple Network Management Protocol. Если перевести дословно, то получится “простой протокол сетевого управления”. Несмотря на название, данный протокол весьма редко используют именно для управления. Наиболее частое применение SNMP - мониторинг. Температура процессора, загрузка канала, свободная оперативная память и так далее. Те же **SNMP Traps**, которые мы обсуждали в пункте 2.1.7. Опять же, мы не будем подробно рассматривать работу этого протокола, т.к. это выходит за рамки нашей книги. Более подробно можно почитать на данном [ресурсе](#). Мы же сосредоточимся на лучших практиках.

Существует три версии протокола: **SNMPv1**, **SNMPv2c** и **SNMPv3**. Не вдаваясь в подробности можно резюмировать, что до появления SNMPv3, главной проблемой SNMP была именно безопасность. Первые две версии протокола имеют очень слабый механизм аутентификации, по сути это лишь один пароль (строка сообщества), который передается в открытом виде. Это весьма серьезная уязвимость, которая позволяет злоумышленнику перехватить этот пароль, после чего он может получить всю необходимую информацию с устройства, на котором запущен SNMP. Если же вы

используете SNMP для управления, то ситуация с безопасностью требует еще большего внимания.

Для решения данной проблемы безопасности был создан протокол SNMPv3, который может использоваться в трех вариантах:

1. **noAuthNoPriv** - пароли передаются в открытом виде, конфиденциальность данных отсутствует;
2. **authNoPriv** - аутентификация без конфиденциальности;
3. **authPriv** - аутентификация и шифрование.

Как вы понимаете, лучше использовать именно третий вариант, который обеспечивает максимальный уровень защищенности.

*Используйте протокол **SNMPv3** с аутентификацией и шифрованием.*

Хотелось бы добавить, что протокол SNMP является важной частью любой корпоративной сети. Он широко применяется для мониторинга всей ИТ-инфраструктуры. Существует даже специализированное программное обеспечение для этих целей (например **Zabbix**), которое автоматически собирает огромное количество информации о всей сети. Эта информация может нести угрозу, если попадет “в руки” злоумышленника. Именно поэтому защищенный вариант SNMP является важным шагом к безопасности вашей сети.

2.5 Резервные копии

Наверняка вы слышали поговорку “Админы делятся на два типа. На тех, кто еще не делает бэкапы, и тех, кто их уже делает”. Поговорка полна иронии, но “в каждой шутке есть доля шутки”.

Очень многие часто забывают про такой важный элемент информационной безопасности, как резервное копирование. Отсутствие бэкапа (от англ. backup) может привести к куда более серьезным последствиям, чем хакерская атака. Системы резервного копирования являются одним из крупнейших сегментов как в области ИБ, так и в области ИТ. Существует целый класс программного обеспечения и оборудования, которое предназначено исключительно для создания бэкапов. Стоимость таких систем может достигать десятки миллионов (и даже не рублей). Рентабельность таких проектов зависит исключительно от стоимости данных, которые могут быть утеряны.

Чаще всего такие дорогостоящие системы применяются для резервного копирования серверов, виртуальных машин, баз данных и т.д. Мы же рассматриваем безопасность сетевых устройств. Все что нам нужно бэкапить, это конфигурации коммутаторов и маршрутизаторов. В данном случае мы сможем обойтись практически подручными средствами без финансовых затрат. При таком раскладе “проблема” резервного копирования всего лишь вопрос желания и ответственности администратора.

В оборудовании Cisco есть два режима бэкапов:

- 1) Регулярные бэкапы. Название говорит само за себя. Бэкапы создаются автоматически через некоторый промежуток времени.

- 2) Бэкапы по команде. Иногда есть необходимость в срочном бэкапе, например перед серьезными изменениями конфигурации. В случае неудачной настройки мы можем практически мгновенно восстановить прежнюю, работоспособную конфигурацию.

Оба режима настраиваются буквально в 5 команд:

```
Router(config)#archive
Router(config-archive)#path flash:backup-config
Router(config-archive)#maximum 14
Router(config-archive)#time-period 1440
Router(config-archive)#write-memory
```

Здесь мы указали, что бэкапы нужно “складывать” на **flash**-память, при этом максимальное количество бэкапов - **14** (затем более новые будут затирать старые). Периодичность бэкапов составляет **1440** минут, что соответствует одному дню. Кроме того, бэкап будет создаваться автоматически при использовании команды **write memory**. Если после этого набрать команду **show archive**, то можно увидеть созданные бэкапы.

Большинство инструкций на этом заканчивается. Однако на мой взгляд приведенных действий недостаточно. Т.к. бэкапы создаются локально, то остается риск потери конфигурации при поломке самого устройства. Т.е. если “сгорит” маршрутизатор, то мы безвозвратно потеряем настройки, что исключает быстрое восстановление сети. Очевидно, что бэкапы нужно хранить централизованно, на выделенном сервере. При этом мы уже обсуждали ранее (пункт 2.4.1), что для таких целей лучше всего подойдет защищенный **SCP-сервер** (вместо традиционного **TFTP**). Это исключит возможность перехвата конфигурации при резервном копировании. Для настройки требуется всего лишь заменить строку с параметром **path**:

```
Router(config-archive)#path scp://user:password@192.168.1.100/Cisco-Conf/$h-$t
```

Обратите внимание на параметры **\$h** и **\$t**. Это так называемые переменные окружения. При их использовании **\$h** автоматически заменится на **hostname** устройства (в данном случае это Router), а **\$t** - на системное время. В итоге мы получим бэкап на SCP-сервере в папке **Cisco-Conf** с именем вида **Router-Apr--8-04-19-44.661-0**. Это весьма удобно, особенно когда на один сервер “льются” бэкапы нескольких устройств. Хотелось бы отметить, что данный функционал доступен в относительно новых прошивках.

Настройте резервное копирование конфигураций.

Следует отметить, что существуют специальные программы для резервного копирования настроек сетевых устройств. Некоторые пишут свои собственные скрипты. Во многом эти средства превосходят описанный метод. Однако у нашего примера есть главное преимущество - простота. От идеи до реализации всего несколько команд.

2.6 Логирование команд

В пункте 1.3.7, который посвящен **AAA**, мы обсуждали **Accounting**. Главная цель использования данной технологии - логирование команд. Т.е. иметь возможность проследить, что именно делал на оборудовании тот или иной пользователь. Однако мы также обсуждали, что использование **AAA-сервера** целесообразно если в сети более 10 устройств. Что же делать администраторам небольших сетей?

Относительно новые версии Cisco IOS позволяют логировать все используемые команды на устройстве. Делается это также с помощью функции **archive**. Кроме того, из логов можно автоматически “вырезать” вводимые пароли, а также отправлять эти логи на **syslog-сервер**. Настройки выглядят следующим образом:

```
Router(config)#archive
Router(config-archive)#log config
Router(config-archive-log-cfg)#logging enable /включаем логирование команд
Router(config-archive-log-cfg)#logging size 200 /задаем количество строк
Router(config-archive-log-cfg)#hidekeys /"прячем" пароли
Router(config-archive-log-cfg)#notify syslog /отправляем на syslog-сервер
```

Теперь с помощью команды **show archive log config all** мы можем видеть кто, как и что именно настраивал:

```
Router#show archive log config all
idx sess user@line Logged command
 1 1 console@console | logging enable
 2 1 console@console | logging size 200
 3 1 console@console | hidekeys
 4 1 console@console | notify syslog
```

В случае отсутствия AAA-сервера настройте логирование вводимых команд.

Хочу заметить, что метод AAA является более универсальным и профессиональным решением. Однако, в некоторых случаях, описанное решение будет более разумным вариантом.

2.7 Обновления

Наверняка вы заметили, что обсуждая некоторые функции безопасности, я периодически упоминаю, что они доступны только в новых версиях прошивок, что уже является весомым поводом для обновления. Однако расширенный функционал не является главной причиной для обновления прошивки. Старое программное обеспечение несет в себе серьезную угрозу для любой сети.

Весьма часто вендор публикует информацию о новых уязвимостях своих продуктов и выпускает новую версию прошивки, либо патч, который должен устранить эти “дыры”. Такие публикации называются “Вендорский бюллетень безопасности”. Более того, на некоторых ресурсах можно найти соответствующие эксплойты (для Exploit-DB или Metasploit), которые можно тут же попробовать для оценки защищенности вашей сети. Для примера можно привести сайт Vulners, который многие называют “Гугл для хакера”.

Как ни странно, но эта информация может быть полезна не только для “честных граждан”. Фактически хакерам сообщают о новых уязвимостях, которые с большой долей вероятности еще не успел никто “закрыть”. Это открывает перед ними широкие возможности. Вы можете поставить самый дорогой межсетевой экран, однако, после обнаружения его уязвимостей и выпуска соответствующих эксплойтов, взломать вашу сеть сможет даже школьник. Именно поэтому очень важно держать все свои системы в максимально актуальном состоянии - самые последние прошивки, патчи и т.д.

Регулярно проверяйте наличие обновлений. У вас должны быть самые “свежие” прошивки.

Регулярно следите за новостями о новых уязвимостях на специализированных ресурсах. Кроме того, на сайте [Vulners](#) можно подписаться на интересующие темы и автоматически получать информацию о новых угрозах (имеется Telegram-бот). Мы не будем подробно рассматривать данный сервис, т.к. эта тема выходит за рамки нашей книги, однако я настоятельно рекомендую ее к [ознакомлению](#).

В заключении данного параграфа мне хотелось бы вспомнить еще один случай из личной практики. Работая у одного из заказчиков я обнаружил коммутатор, который работал без перезагрузок (uptime) целых 11 лет. Это конечно великолепный показатель для оборудования - надежность. Однако за эти 11 лет используемая прошивка уже перестала поддерживаться, а количество уязвимостей исчисляется десятками. Не делайте подобных ошибок.

2.8 Чек-лист №2

Вторая глава подошла к концу, резюмируем все вышесказанное в виде Чек-листа №2:

- 1) *Информационная безопасность невозможна без использования логов.*
- 2) *Используйте несколько способов сбора логов.*
- 3) *С осторожностью используйте повышенные уровни логирования.*
- 4) *Используйте **Console Logging** только при необходимости.*
- 5) *Настройте **Buffered Logging** даже при использовании **Syslog-сервера**.*
- 6) *Используйте **Terminal Logging** с осторожностью и только в случае острой необходимости.*
- 7) *Используйте Лог-серверы для централизованного сбора логов.*
- 8) *Уделите особое внимание защите **Syslog-сервера**.*
- 9) *Не забывайте про настройку **NTP** и **Timestamps**.*
- 10) *Отключите неиспользуемые сервисы.*
- 11) *Зарезервируйте оперативную память для консольного подключения.*
- 12) *Используйте **SCP** вместо **TFTP**.*
- 13) *Используйте протокол **SNMPv3** с аутентификацией и шифрованием.*
- 14) *Настройте резервное копирование конфигураций.*
- 15) *В случае отсутствия **AAA-сервера** настройте логирование вводимых команд.*
- 16) *Регулярно проверяйте наличие обновлений. У вас должны быть самые “свежие” прошивки.*

2.9 Пример конфигурации

1) Настройка **Buffered Logging** и уровня логирования

```
Router(config)# logging on
Router(config)# logging buffered 32768
Router(config)# logging buffered informational
```

2) Настройка **Syslog**-сервера и уровня логирования

```
Router(config)#logging host 192.168.1.100
Router(config)#logging trap informational
```

3) Ограничение количества логов с одного устройства

```
Router(config)#logging rate-limit all 50
```

4) Настройка времени

```
Router(config)#ntp server 192.168.1.100
Router(config)#clock timezone MSK 4
Router(config)#service timestamps log datetime msec localtime show-timezone
```

5) Резервирование оперативной памяти для консольного подключения

```
Router(config)#memory reserve console 4096
```

6) Настройка резервного копирования конфигурации на SCP-сервер

```
Router(config)#archive
Router(config-archive)#path scp://user:password@192.168.1.100/Cisco-Conf/$h-$t
Router(config-archive)#maximum 14
Router(config-archive)#time-period 1440
Router(config-archive)#write-memory
```

7) Настройка логирования команд (в случае отсутствия AAA-сервера)

```
Router(config)#archive
Router(config-archive)#log config
Router(config-archive-log-cfg)#logging enable
Router(config-archive-log-cfg)#logging size 200
Router(config-archive-log-cfg)#hidekeys
Router(config-archive-log-cfg)#notify syslog
```

8) Отключение "лишних" сервисов

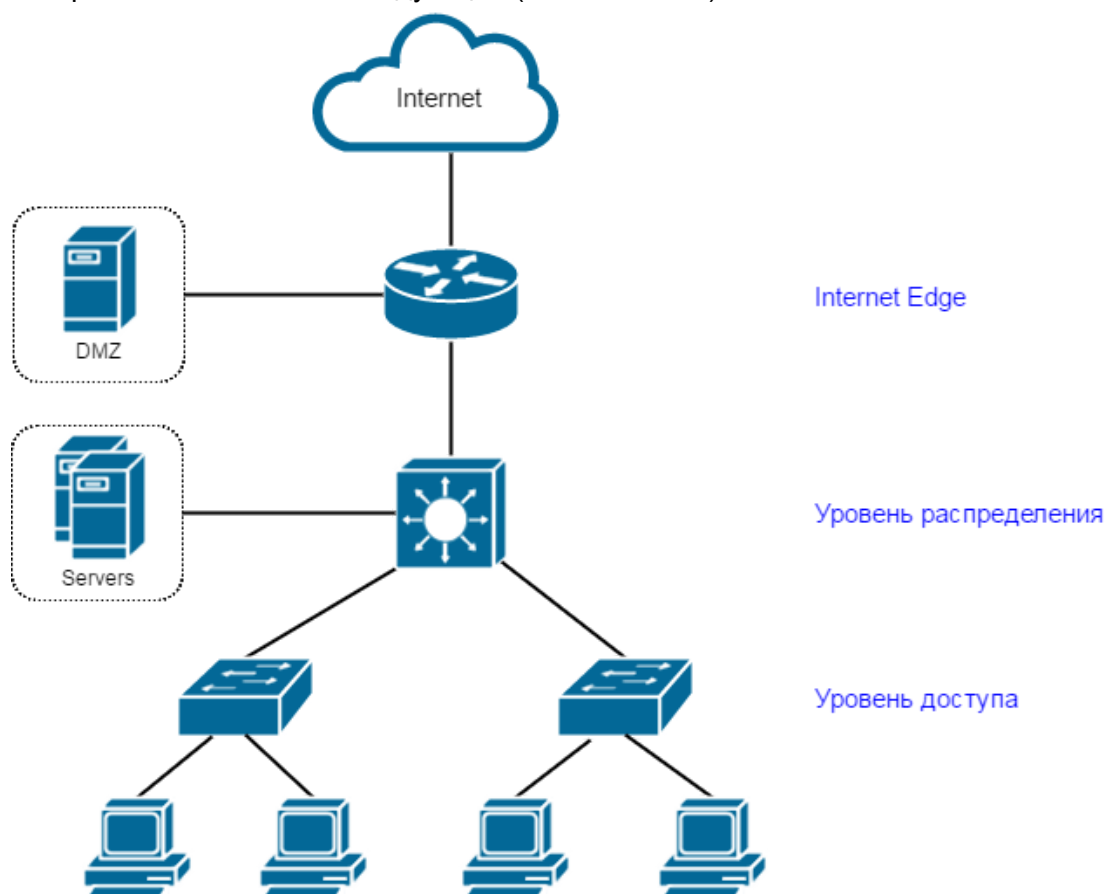
```
no service tcp-small-servers echo
no service tcp-small-servers discard
no service tcp-small-servers daytime
no service tcp-small-servers chargen
no service udp-small-servers echo
no service udp-small-servers discard
no service udp-small-servers daytime
no service udp-small-servers chargen
no ip finger
no ip bootp server
no ip dhcp boot ignore
no service dhcp
no mop enabled
no ip domain-lookup
```

```
no service pad
no ip http server
no ip http secure-server
no service config
no cdp enable
no cdp run
no lldp transmit
no lldp receive
no lldp run global
```

3. Защищаем локальную сеть

Наконец-то мы добрались до основной части книги. Уверен, что примеры из этой главы покажутся читателю наиболее интересными. Два предыдущих раздела были некой подготовительной площадкой, опираясь на которую можно приступить к изучению наиболее значимых (с точки зрения безопасности сетей) вещей. Однако, это ни в коем случае не значит, что можно пренебрегать описанными ранее рекомендациями, которые являются обязательным дополнением к комплексной защите.

Очень трудно говорить о методах защиты, при этом не опираясь на конкретный пример сети. Начиная с этого момента все описываемые в книге методы будут рассматриваться на основе следующей (классической) схемы:



Это схема сильно упрощена, однако большинство описанных методов можно и нужно применять в более крупных и сложных сетях. При этом, рассматривая защиту сети можно выделить два основных пласта:

- 1) Защита в локальной сети;
- 2) Защита периметра (т.е. на границе с сетью Интернет).

В этой главе мы рассмотрим первый вариант. Как я уже говорил ранее, опасность для сети может нести не только злоумышленник-хакер, но и обычные пользователи, которые представляют не меньшую угрозу для работоспособности ИТ-инфраструктуры. Именно поэтому защищенность локальной сети от внутренних угроз имеет столь высокий приоритет.

Несмотря на всю очевидность и реальную опасность угроз (которые будут описаны) большинство системных администраторов или сетевых инженеров продолжают упорно игнорировать их. А если учитывать тот факт, что для защиты от этих угроз нужно ввести всего несколько команд на сетевом оборудовании, то становится очевидной непростительная халатность либо необразованность специалиста. Автор заранее извиняется за столь резкое заявление и искренне надеется, что данная книга будет полезна тем, кто действительно интересуется информационной безопасностью и ищет способы эффективной защиты своей сети, даже без наличия специализированных средств защиты (межсетевые экраны, IPS и т.д.).

3.1 Уровень доступа

Читая различные форумы (а иногда и личные сообщения от подписчиков блога netskills) очень часто можно наткнуться на темы подобного рода:

- Все время падает сеть.
- Пользователи по DHCP получают чужие IP-адреса. Что делать?
- Юзеры ставят свои хабы. Как запретить?
- Кто-то спуфит сетку.
- Упала сеть после подключения коммутатора.

Все эти проблемы также являются предметом информационной безопасности. Самое интересное, что доля таких “инцидентов” гораздо больше, чем доля реальных хакерских атак. Главная причина - неверная настройка самого обычного сетевого оборудования (коммутаторов и маршрутизаторов). При этом защиту локальной сети стоит рассматривать последовательно и начинать с самого низа - уровня доступа. Именно на этом уровне больше всего проблем и потенциальных угроз.

3.1.1 Сегментация

Сегментация - первое с чего начинается любая защищенная сеть. Многие относятся к сегментам исключительно как к инструменту безопасности. На самом деле сегментация нужна для создания структуры и порядка. Подобные структуры из сегментов окружают нас повсюду - страны, улицы, дома, парковые зоны и так далее. Без порядка наступает хаос, а хаос это самая опасная угроза для любой сети и не только.

Наверняка читатель знает, что такое широковещательный (**broadcast**) домен - именно его мы будем называть сегментом. Но многие забывают про то, что чем больше этот домен, тем больше потенциальных угроз. Фактически, каждый порт коммутатора является угрозой для сегмента которому он принадлежит. И чем больше портов в сегменте, тем больше шансов “получить” проблему. Поэтому даже если перед вами нет задачи защитить свою сеть от каких-то умышленных угроз (например хакерских атак), не стоит пренебрегать сегментацией. Сегментация позволит вам значительно повысить стабильность сети в целом. Даже в случае возникновения проблемы в каком-то сегменте, все последствия локализуются внутри, при этом остальная сеть продолжает работать в штатном режиме.

Существует два основных способа сегментации:

- 1) **Физическая.** В этом случае разные сегменты физически используют разное оборудование (коммутаторы или маршрутизаторы). Применяется не так часто (из-за стоимости) и только в очень критических инфраструктурах.
- 2) **Логическая.** Самый распространенный вариант. Сеть может быть сегментирована даже в рамках одного коммутатора. Достигается это за счет использования технологии VLAN.

Уверен, что читающие эту книгу в полной мере понимают, что такое VLAN и для чего он нужен. Если же кто-то имеет пробел в данной теме либо сомневается в своих знаниях, то может посмотреть [5-й](#) видео урок из "[Курса молодого бойца](#)" где я частично рассказываю об этом. Мы же не будем подробно останавливаться на этой теме.

Избегайте "больших" широкоэвещательных доменов. Сегментируйте сеть для создания структуры и порядка.

У многих часто возникает вопрос: "По какому принципу сегментировать сеть? Что выделять в отдельный сегмент?". Приведу несколько примеров:

- 1) Сегментация по отделам компании (бухгалтерия, администрация, отдел кадров, пользователи, гостевой сегмент).
- 2) Выделение серверов в отдельный сегмент (Active Directory, файловое хранилище, корпоративный портал).
- 3) Выделение DMZ (публичные сервисы). Об этом поговорим чуть позже.
- 4) Сегмент для управления. Мы говорили об этом в пункте 1.2.5.

Это всего лишь несколько примеров из большого множества. Сегментация появляется там, где есть разные роли (как пользователей, так и серверов). Только так можно получить контроль над трафиком и уменьшить вероятность сбоев всей сети. Кроме того, сегментированная сеть гораздо проще масштабируется и модернизируется. На моей практике было много случаев, когда добавление нового сервиса или средства защиты превращалось в невыполнимую задачу, т.к. сеть была не сегментирована и завершение проекта требовало полного переделывания сетевой архитектуры.

Лишь при совпадении двух условий можно отказаться от сегментации в локальной сети:

- 1) "Большой" трафик - сотни мегабит между участниками локальной сети.
- 2) Отсутствие коммутатора третьего уровня (L3 switch) либо "мощного" маршрутизатора.

При совпадении этих условий сегментация может стать невыполнимой задачей. Разделив сеть на несколько VLAN-ов придется маршрутизировать трафик между ними. И если трафик "большой", а маршрутизирующее устройство "слабое", то встанет проблема с работоспособностью сети. Поэтому будьте внимательны и продумывайте такие вещи заранее. В идеальном случае после уровня доступа должен следовать уровень распределения или уровень ядра, где ставится L3 коммутатор (подробнее об этом можно почитать в книге "[Архитектура корпоративных сетей](#)"). Но мы живем не в идеальном мире и зачастую после уровня доступа сразу следует пограничный маршрутизатор, который должен справляться с необходимой нагрузкой. В одной из статей в блоге я уже писал [чем отличается L3 коммутатор от маршрутизатора](#) (настоятельно рекомендую к ознакомлению).

В последние годы набирает популярность идея “микросегментации”. Особенно когда дело касается виртуальной инфраструктуры и SDN сетей. Микросегментация позволяет получить максимальный контроль над трафиком, однако существенно усложняет конфигурацию и администрирование. У этого подхода есть свои плюсы и минусы, которые мы не будем обсуждать в рамках этой книги.

3.1.2 Неиспользуемые порты

Одна из самых распространенных ошибок при настройке коммутаторов доступа (к которым подключаются пользователи, сервера, ip-камеры и т.д.) - НЕ отключение НЕиспользуемых портов. Типичная ситуация, когда “свободные” порты оставляют включенными.

В чем опасность? Очевидно, что при наличии физического доступа к коммутатору, любой может подключиться и тут же получить доступ к корпоративной сети. Этот факт усугубляется тем, что большинство администраторов оставляют порты с настройками по умолчанию. “Дефолтные” настройки подразумевают, что порт находится в **VLAN 1** и на нем включен протокол **DTP** (switchport mode dynamic desirable), что в свою очередь позволяет злоумышленнику получить доступ и к другим сегментам сети (чем опасны VLAN1 и протокол DTP мы поговорим чуть позже). Кроме того нерадивые пользователи любят подключать свои хабы или свичи именно к “свободным” портам, нередко создавая сетевую “петлю”. Учитывая возможные проблемы, правилом хорошего тона является отключение неиспользуемых портов, либо их перевод в нигде неиспользуемый VLAN.

Отключайте неиспользуемые порты, либо переводите их в неиспользуемый VLAN.

Отключение порта производится с помощью команды **shutdown** в режиме конфигурации соответствующего интерфейса:

```
Switch(config)#interface fa0/24  
Switch(config-if)#shutdown
```

Если необходимо отключить сразу несколько портов, то можно воспользоваться следующей командой:

```
Switch(config)#interface range fa0/21 - 24  
Switch(config-if)#shutdown
```

Как видите, всего лишь две строчки могут существенно повысить безопасность вашей сети.

3.1.3 VLAN1

Почти уверен, что большинство читающих эту книгу слышали фразу “Не используйте VLAN 1”. Однако, далеко не все могут объяснить причину. Попробуем понять в чем же опасность.

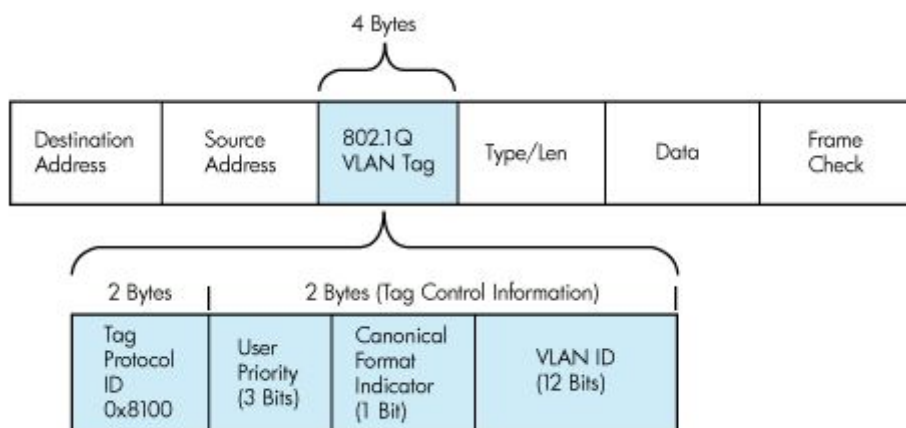
Самый главный недостаток первого VLAN-а в том, что в большинстве коммутаторов он является VLAN-ом по умолчанию. Это значит, что если не трогать настройки коммутатора и подключиться к любому из его портов, то мы автоматически оказываемся в VLAN 1. Такая ситуация довольно часто наблюдается в небольших

сетях, где всего-лишь один или два сегмента, а VLAN 1 оставляется для пользователей. Это весьма плохая практика, от которой нужно избавляться, как от вредной привычки. Выше мы уже рассмотрели опасность включенных неиспользуемых портов, которая тесно связана с настройками по умолчанию. Ведь если не использовать VLAN 1, то даже при успешном подключении к неиспользуемому порту, злоумышленник не сможет получить доступ к корпоративной сети, т.к. будет находиться в первом “влане”.

Не используйте VLAN 1.

Данный совет можно отнести к так называемым “Best practice”. Придерживайтесь данной рекомендации и заранее планируйте свою сеть, исключая использование VLAN 1. Фактически, не стоит оставлять порты коммутатора с настройками по умолчанию. Всегда определяйте порт в конкретный VLAN (если это не trunk порт).

Однако это не единственная причина для отказа от VLAN 1. Есть еще одна неочевидная проблема. Проблема кроется в нечетком описании стандарта **IEEE 802.1Q** (открытый стандарт, который описывает процедуру тегирования трафика для передачи информации о принадлежности к VLAN). На рисунке ниже показано, как происходит вставка тега **802.1Q** в кадр **Ethernet-II**.



Как можно заметить тег занимает 4 байта и включает в себя 4 поля: **Tag Protocol ID**, **User Priority**, **Canonical Format Indicator** и **VLAN ID**. VLAN ID имеет размер 12 бит, а значит он может варьироваться в диапазоне от 0 до 4095 (2 в степени 12). Вставка подобного тега (тегирование) происходит всякий раз, когда трафик входит в **access** порт.

Говоря фразу “**VLAN по умолчанию**” имеется ввиду, что с “дефолтными” настройками порта весь входящий трафик тегруется в первый VLAN (т.е. в поле VLAN ID устанавливается “1”). Однако данное правило действует далеко не для всех коммутаторов. В некоторых свичах “дефолтным” VLAN-ом является “0”, либо тег вообще не вставляется. В результате могут возникнуть весьма “странные” эффекты при “стыковке” коммутаторов с разными VLAN-ми по умолчанию. Чтобы не сталкиваться с этими трудностями следует отказаться от использования VLAN 1 и портов с настройками по умолчанию.

Чтобы изменить VLAN на **Access** порту достаточно ввести всего две команды:

```
Switch(config)#interface fa0/24
```

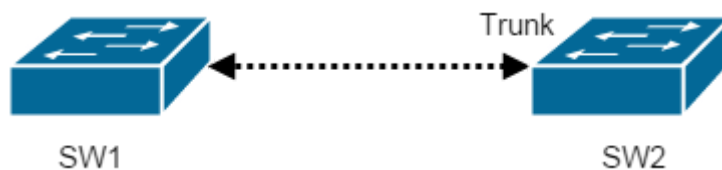
```
Switch(config-if)#switchport access vlan 2
```

Для группы портов используйте range

```
Switch(config)#interface range fa0/21 - 24  
Switch(config-if)#switchport access vlan 2
```

3.1.4 DTP

В коммутаторах Cisco имеется весьма интересный протокол **DTP** - dynamic trunking protocol (динамический протокол транкинга). Это проприетарный (собственная разработка) протокол, который позволяет коммутаторам динамически распознавать настроен ли соседний коммутатор для поднятия транка и какой протокол использовать (**802.1Q** или **ISL**). Самое главное, что данный протокол включен по умолчанию, а значит любой порт может перейти в состояние Trunk, если будет подключен к другому Trunk-порту. Рассмотрим на примере:



Если на коммутаторе **SW2** настроить Trunk-порт и подключить его к **SW1**, то порт первого коммутатора автоматически перейдет в состояние Trunk и будет передавать все имеющиеся на нем VLAN-ы. С одной стороны это очень удобно, т.к. нет необходимости настраивать оба коммутатора. Но с другой стороны, если вместо SW2 подключится компьютер злоумышленника, то никто не мешает ему настроить свой порт как Trunk и “вытянуть” все VLAN-ы с вашего коммутатора. Данная ситуация возможна из-за особенности работы DTP протокола. Рассмотрим его чуть подробнее. При работе DTP, порт коммутатора может находиться в трех режимах:

- 1) **auto** - Порт находится в автоматическом режиме и будет переведен в состояние Trunk, только если порт на другом конце находится в режиме on или desirable. Т.е. если порты на обоих концах находятся в режиме "auto", то trunk применяться не будет.
- 2) **desirable** - Порт находится в режиме "готов перейти в состояние Trunk"; периодически передает DTP-кадры порту на другом конце, запрашивая удаленный порт перейти в состояние Trunk (состояние Trunk будет установлено, если порт на другом конце находится в режиме on, desirable, или auto).
- 3) **nonegotiate** - Порт готов перейти в состояние Trunk, но при этом не передает DTP-кадры порту на другом конце. Этот режим используется для предотвращения конфликтов с другим "не-cisco" оборудованием. В этом случае коммутатор на другом конце должен быть вручную настроен на использование Trunk.

В более старых моделях коммутаторов по умолчанию включен режим **auto**, а в более новых - **desirable**. В целом, это не имеет значения, т.к. оба режима могут быть использованы хакером для создания Trunk - соединения. Именно поэтому рекомендуется отключать неиспользуемые порты. Однако этого недостаточно, ведь злоумышленник может воспользоваться портом пользователя (просто переткнуть кабель). Чтобы исключить возможность автоматического переключения порта в Trunk,

необходимо вручную переключить его в состояние **Access**. Делается это весьма просто:

```
Switch(config)#interface fa0/23
Switch(config-if)#switchport mode access
```

Очень многие администраторы забывают про эту команду, что существенно повышает риски для сети. Вообще говоря, настройку любого порта нужно начинать именно с определения режима порта, будь то Access или Trunk. Только после этого можно переходить к другим конфигурациям порта. Это должно стать для вас привычкой, которая выполняется на автомате. Аналогичная ситуация с Trunk - портами:

```
Switch(config)#interface fa0/23
Switch(config-if)#switchport mode trunk
```

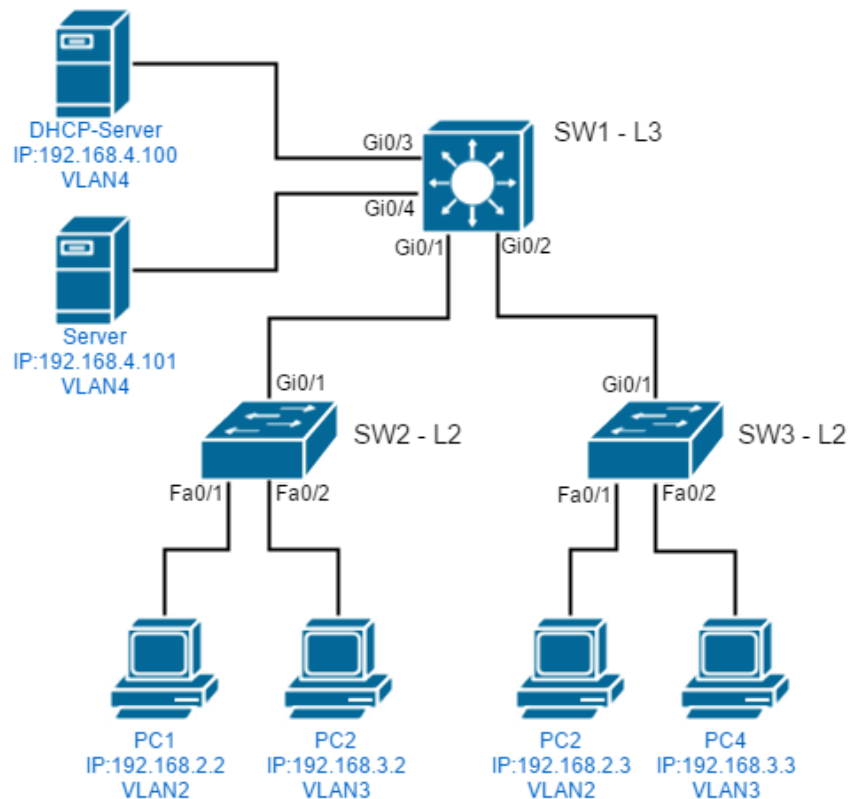
*Все порты должны иметь строго определенный режим: **mode access** или **mode trunk**.*

3.1.5 Trunk - порты

Одной из самых распространенных ошибок при настройке Trunk - портов является разрешение абсолютно всех VLAN-ов. Довольно часто можно встретить настройки подобного вида:

```
interface FastEthernet0/24
switchport mode trunk
```

В такой конфигурации Trunk-порт будет передавать все VLAN-ы, которые имеются на данном коммутаторе. С одной стороны это удобно, т.к. настройка “транка” ограничивается всего лишь одной командой. Однако есть и существенные минусы, которые в первую очередь касаются безопасности и стабильности сети. Рассмотрим стандартную схему сети:



Как видим, серверный сегмент находится в отдельном VLAN, доступ к нему возможен лишь через L3 устройство, на котором, как правило, настроены списки доступа (access-list). Однако если в Trunk - порте разрешены абсолютно все VLAN-ы, то для пользователей теоретически появляется возможность получить доступ в обход маршрутизирующего устройства. Если злоумышленник смог получить управление над коммутатором доступа, то он сможет обнаружить какой тег используется для серверного VLAN-а. После этого получить прямой доступ к серверному сегменту не составит труда, т.к. Trunk - порт пропускает все VLAN-ы.

Что касается стабильности сети, то нередко подобная настройка приводит к образованию логических “петель”, т.е. когда закольцовываются VLAN. Это происходит особенно часто, когда между коммутаторами существует несколько линков.

Чтобы избежать подобных проблем настоятельно рекомендуется указывать какие именно VLAN-ы разрешены в Trunk-порте. Это правило также относится к категории “best practice” и должно стать хорошей привычкой. Для указания разрешенных VLAN-ов используйте следующую команду:

```
Switch(config)#interface fa0/24
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 2,3,10-14,20
```

Для добавления VLAN в Trunk - порт используйте команду:

```
Switch(config-if)#switchport trunk allowed vlan add 100
```

Бывают ситуации, когда VLAN-ов очень много и проще запретить несколько, чем перечислять все разрешенные:

```
Switch(config-if)#switchport trunk pruning vlan 5
```

*Не разрешайте все VLAN-ы в Trunk - портах. С помощью команды **switchport trunk allowed vlan** указывайте конкретные VLAN-ы.*

3.1.6 Имена портов и VLAN

Выше было упомянуто, что без порядка наступает хаос, а хаос это самая большая угроза для любой сети. И это не просто слова. Стоит признать, что большинство администраторов пренебрегают документированием ИТ - инфраструктуры и довольно часто под рукой не оказывается даже актуальной схемы сети. В этом случае настройка осуществляется “по памяти”, что нередко приводит к ошибкам. А когда возникает ошибка, то искать проблему без понимания сети еще труднее (особенно если ваша сеть больше, чем один коммутатор). Множество раз я наблюдал картину, когда из-за отсутствия описания портов происходили ошибки конфигурирования или процесс “траблшутинга” сети затягивался на часы, а то и дни.

Чтобы избежать подобных проблем следует придерживаться еще одной рекомендации из категории “best practice”:

*Указывайте **description** для ключевых портов коммутатора и давайте имена всем создаваемым VLAN-м.*

Рассмотрим данную рекомендацию чуть подробнее. Что имеется ввиду под “ключевыми” портами коммутаторов? В первую очередь это порты, к которым подключены другие коммутаторы и порты выделенные для серверов. Т.е. по сути, это самые важные порты, необходимые для работы сети.

Пример настройки порта, к которому подключается другой коммутатор:

```
SW1(config)#interface fa0/24
SW1(config-if)#description Link_to_SW2
SW1(config-if)#switchport mode trunk
SW1(config-if)#switchport trunk allowed vlan 2,3,10-14,20
```

Пример настройки порта, к которому подключается сервер виртуализации:

```
SW1(config)#interface fa0/20
SW1(config-if)#description SRV_ESXi
SW1(config-if)#switchport mode access
SW1(config-if)#switchport access vlan 10
```

С помощью **description** мы можем однозначно определить предназначение того или иного порта, что уменьшает шансы совершения ошибки и существенно сокращает время “траблшутинга”.

Что касается портов пользователей, то большого смысла в их наименовании нет. Достаточно знать, что это пользовательский порт, о чем может свидетельствовать отсутствие **description** и **mode access**. Кроме того в больших сетях весьма трудно поддерживать актуальную информацию о подключении обычных компьютеров ввиду динамичности изменений (подключение новых пользователей, переезд в другую комнату и т.д.), в то время как ключевые порты большую часть времени статичны.

Однако, для небольших сетей именованние портов весьма реальная задача, которой не стоит пренебрегать.

В этом же параграфе хотелось обсудить не менее важный вопрос - имена для VLAN-ов. Здесь абсолютно аналогичная ситуация. Отсутствие имен и настройка “по памяти” почти всегда ведет к проблемам и простою сети, особенно когда количество VLAN-ов велико. Перед применением “влана” на порт, для начала следует его создать и задать имя. Это своего рода правило хорошего тона. Имена VLAN-ов должны быть осмысленными и однозначно определять их предназначение. Процедура создания VLAN-а выглядит следующим образом:

```
Switch(config)#vlan 10
Switch(config-vlan)#name SRV
```

Описанные рекомендации помогут избежать хаоса и должны стать полезной привычкой для любого администратора сетевого оборудования.

3.1.7 Штормы

В книге уже не раз упоминалось, что внутренние пользователи представляют опасность для сети, не меньше, чем хакеры. Связано это в первую очередь с неосторожными действиями. Именно поэтому в самом начале было рекомендовано обеспечить физическую защиту сетевого оборудования. Однако это не всегда возможно. И самая частая проблема, которая может вытекать из этого - широковещательный шторм (“петля”).

Для проверки защищенности сети от штормов необходим всего лишь патчкорд. С помощью него необходимо соединить два соседних порта коммутатора (либо соседние сетевые розетки). Это вполне типичная ситуация для офисов, когда нерадивые пользователи организуют подобные “петли”. Если сетевое оборудование с “дефолтными” настройками, то начинается лавинный рост широковещательного трафика, что приводит к значительной деградации пропускной способности сети. Фактически сеть просто перестает работать. Мы не будем подробно рассматривать механизм широковещательного шторма (более подробно [ТУТ](#)), но мы рассмотрим способ защиты от него. Для предотвращения “петли” на определенном порте необходимо использовать следующие команды:

```
Switch(config)#int fa0/20
Switch(config-if)#storm-control broadcast level 10.00
Switch(config-if)#storm-control action shutdown
```

В данном примере мы устанавливаем максимальный уровень широковещательного (**broadcast**) трафика в 10% от полосы пропускания. Порог можно устанавливать не только в процентах, но и в битах в секунду (bps). Последняя команда определяет, какое действие должно быть совершено при достижении лимита - отключить порт (**shutdown**). Если действие не указывать, то свитч будет просто отбрасывать трафик, превышающий порог. С помощью **action trap** можно настроить информирование о “петле”: SNMP-трапы, Syslog. Для автоматического восстановления порта из shutdown состояния можно использовать следующие команды:

```
Switch(config)#errdisable recovery cause storm-control
Switch(config)#errdisable recovery interval 300
```


В данном случае интервал установлен в 300 секунд, т.е. 5 минут.

При необходимости настройте автоматическую защиту от широкоэвещательных штормов.

У коммутаторов Cisco есть также штатное средство обнаружения петель, основанное на периодической отправке keeralive - сообщений. Эта опция обычно включена по умолчанию, и в случае срабатывания порт отключается. Для механизма keeralive можно также настроить автоматическое включение интерфейса:

```
Switch(config)#errdisable recovery cause loopback
```

3.1.8 PortFast

PortFast - еще одна полезная функция о которой забывают многие администраторы сети. Наверняка большинству известно, что протокол **STP** (spanning tree protocol) включен по умолчанию на всех коммутаторах Cisco. STP это стандартизированный протокол защиты от "петель". Мы не будем подробно рассматривать его работу, т.к. эта тема выходит за рамки нашей книги. Однако, ниже вы встретите большое количество специфических терминов, поэтому настоятельно рекомендуется "[освежить](#)" свои познания в STP, прежде чем продолжать чтение.

По умолчанию, при включении интерфейса коммутатора, порт в обязательном порядке проходит три этапа: **listening**, **learning** и уже после этого STP переводит порт либо в состояние **forwarding**, либо **blocking**. Данная процедура занимает около 30 секунд и необходима коммутатору для понимания, что к нему подключили - компьютер или еще один "свич"? Весьма часто возникает ситуация, когда компьютер пользователя загружается менее чем за 10 секунд и начинает пытаться получить ip-адрес по DHCP протоколу. Но порт все еще не "поднялся" и в результате компьютер, не получив ничего по DHCP, выполняет автонстройку, выбирая адрес типа **169.254.x.x**. Для избежания подобных ситуаций используется функция PortFast.

Фактически, PortFast меняет две вещи в стандартной работе STP:

- 1) порт пропускает состояния listening и learning;
- 2) при изменении статуса порта, не отправляется сообщение о изменении состояния порта TCN BPDU (topology change notification BPDU).

Это позволяет практически моментально переводить порт в состоянии forwarding при подключении патчкорда. Кроме того, выключение TCN BPDU позволяет снизить нагрузку на Root Bridge (корневой коммутатор STP), которому пришлось бы обрабатывать каждое включение/отключение портов. Это весьма заметно в больших L2 сетях.

*Активируйте функцию **PortFast** на портах пользователей.*

Использовать PortFast рекомендуется только на портах, к которым подключены конечные станции: компьютеры, сервера, ip-камеры и так далее. Применение PortFast на портах, к которым подключены коммутаторы, может привести к образованию

“петель”, т.к. в этом случае STP не отрабатывает в полной мере. Настройка PortFast выполняется следующим образом:

```
Switch(config)#int range fa0/1 - 20
Switch(config-if-range)#spanning-tree portfast
```

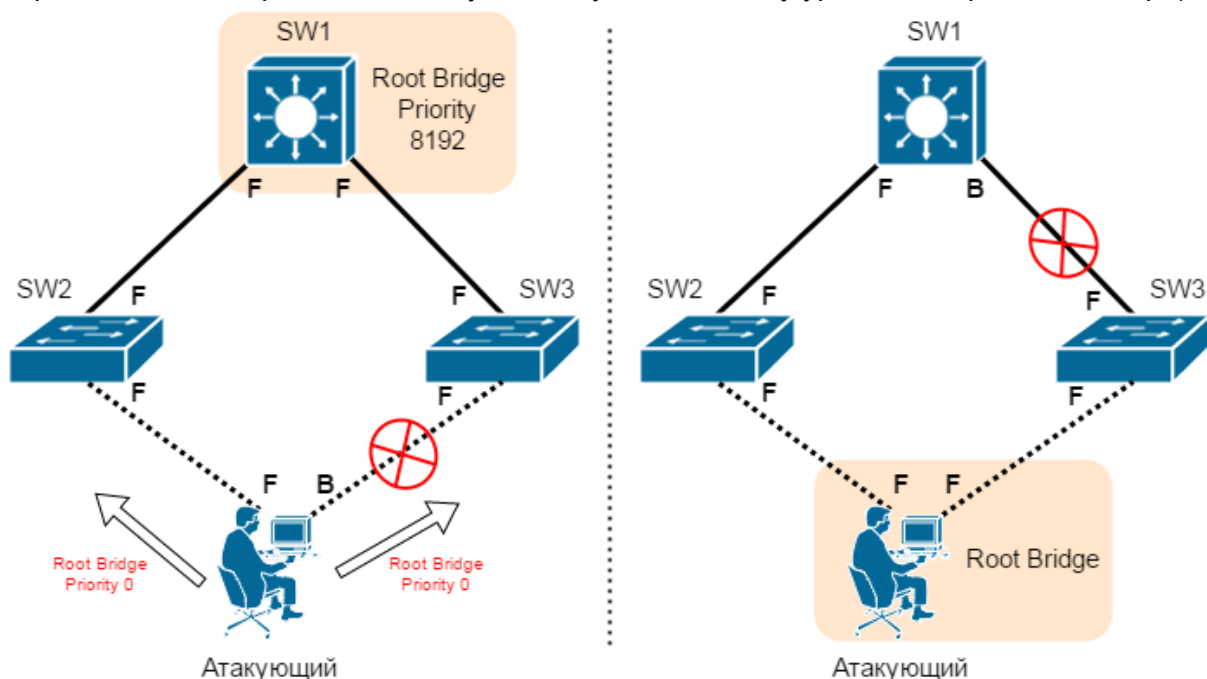
В данном случае мы включили PortFast сразу для диапазона портов. Есть другой способ, который из глобальной конфигурации включает PortFast на всех Access-портах:

```
Switch(config)#spanning-tree portfast default
```

Используйте данную команду только если вы уверены, что ко всем Access-портам подключены конечные устройства, а не коммутаторы.

3.1.9 BPDU Guard

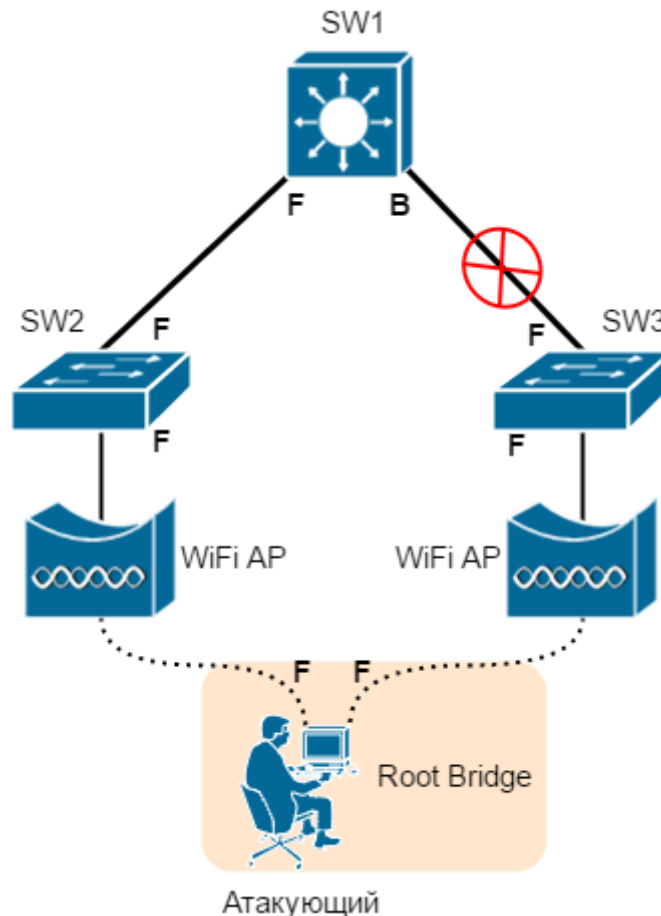
Рассмотрим один из самых распространенных сценариев взлома локальных сетей - подмена корневого коммутатора. На рисунке ниже представлена типовая схема сети - два L2 коммутатора доступа, которые подключены к L3 коммутатору распределения (если у вас трудности с пониманием данных терминов, то рекомендую к прочтению первую часть книги [“Архитектура корпоративных сетей”](#)). В данном случае SW1 является корневым (**Root Bridge**), что весьма логично. При этом корневой коммутатора выбирается на основе минимального значения **Bridge Priority** (8192 в нашем случае). В случае если коммутаторы доступа (SW2 и SW3) имеют “дефолтные настройки”, то для хакеров появляется потенциальная возможность подмены корневого коммутатора. На картинке ниже представлен процесс атаки (мы еще вернемся к этой картинке, когда будем обсуждать защиту уровня дистрибуции/ядра).



Атакующему требуется подключиться к SW2 и SW3. Он может использовать собственный коммутатор, либо компьютер с двумя сетевыми интерфейсами. По умолчанию один из его линков будет заблокирован протоколом STP, т.к. SW1 является

корневым. Однако злоумышленник может начать рассылать BPDU пакеты с нулевым значением Bridge Priority. В результате STP топология будет перестроена и устройство атакующего будет выбрано в качестве корневого коммутатора. Это позволит хакеру перехватывать весь трафик пользователей, подключенных к одному из коммутаторов (в нашем случае - SW3). Получается атака типа **Man in the middle (MITM)**.

Многие администраторы считают, что данную атаку невозможно выполнить в их сети, т.к. коммутаторы доступа расположены на большом расстоянии друг от друга и атакующий просто не сможет создать два таких линка. Это большое заблуждение, т.к. существует способ обхода подобных физических ограничений - беспроводные технологии. На рисунке ниже приводится пример.



К коммутаторам SW2 и SW3 подключены точки доступа, которые работают в режиме Bridge. Это позволяет атакующему организовать подмену корневого коммутатора посредством беспроводных линков.

Организовать защиту от подобных атак довольно просто. Для этого требуется воспользоваться функцией **BPDU Guard** на всех пользовательских портах (исключая порты, к которым подключены коммутаторы). Это позволит блокировать BPDU пакеты там, где их и не должно быть.

*Используйте **BPDU Guard** на пользовательских портах.*

Активировать данную защиту можно для отдельно порта:

```
Switch(config)#int fa0/1
```

```
Switch(config-if)#spanning-tree bpduguard enable
```

Но гораздо логичнее включить данную функцию глобально для всех пользовательских портов (на которых уже сконфигурирован PortFast):

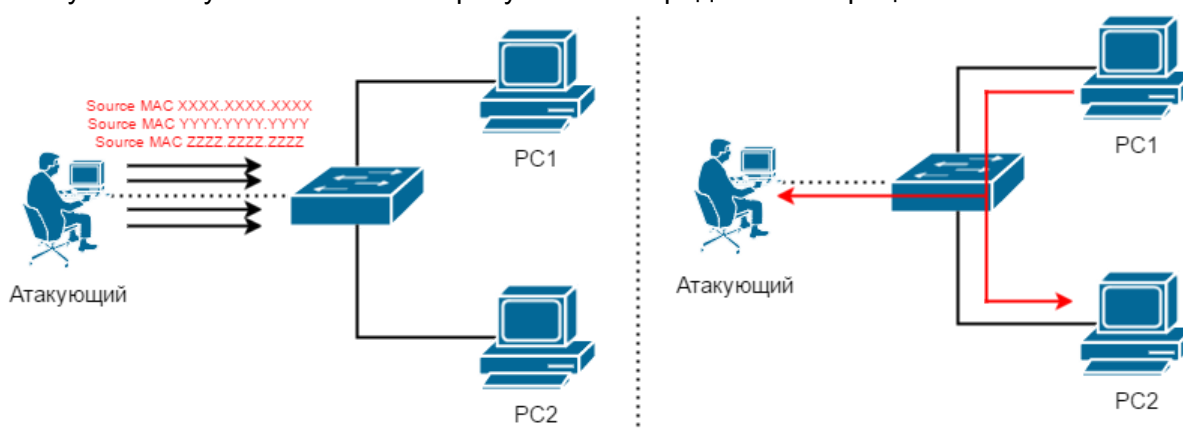
```
Switch(config)#spanning-tree portfast bpduguard default
```

BPDU Guard в связке с PortFast позволяет существенно повысить защищенность и стабильность сети.

3.1.10 Port Security

Начать данный параграф мне бы хотелось с небольшой истории. На одном из прошлых мест работы мне довелось организовывать всю локальную сеть. В компании был отдел монтажников, которым полагался один компьютер с выходом в интернет. В то время я еще даже не задумывался о безопасности. Спустя какое-то время мы стали замечать серьезные “тормоза” в работе сети. Проведя небольшое “расследование” обнаружили, что монтажники подключили свой коммутатор, а к нему еще порядка 10-и собственных ноутбуков. Свои устройства они использовали не для работы, а для игры в World of Tanks и скачивания фильмов через Torrent. Естественно это серьезно сказывалось на общей “скорости” Интернет. Кроме того, на многих ноутбуках были обнаружены вирусы, из-за которых было “зашифровано” общее дисковое хранилище.

К этой же теме относится еще один вид распространенных атак - **Mac-flooding**. Ее смысл заключается в переполнении CAM таблицы коммутатора - таблица соответствия порта и MAC-адреса. Для примера, коммутатор Cisco 2960 может содержать 8024 записей MAC-адресов. Т.е. размер конечный и именно этим пользуются злоумышленники. На рисунке ниже представлен процесс атаки.



Атакующий подключается к коммутатору и с помощью специального программного обеспечение генерирует на порту тысячи MAC-адресов. Коммутатор сохраняет все в CAM-таблицу, до ее переполнения. Таким образом хакер вытесняет или затирает все легитимные MAC-адреса. После переполнения таблицы коммутатора переходит в режим fail open mode, т.е. фактически превращается в обычный “хаб”. А это значит, что каждый входящий пакет будет рассылаться по всем портам. Таким образом, при сетевом взаимодействии PC1 и PC2, атакующий сможет видеть весь трафик.

Данный тип атаки легко смоделировать даже в сетевых эмуляторах (GNS3/UNetLab/EVE). Для этого вам понадобится коммутатор и дистрибутив Kali Linux, в состав которого входит нужная утилита - **MacOf**.

Для защиты от описанных угроз настоятельно рекомендуется включать функцию **Port Security** на всех пользовательских портах. Таким образом вы пресечете возможность использования “нелегальных” коммутаторов и сможете защитить сеть от атак типа Mac-flooding.

*Используйте **Port Security** на всех пользовательских портах.*

Кроме того, вы сможете защититься от подмены MAC-адреса с целью перехвата трафика. К примеру MAC-адрес файлового сервера будет разрешен только на соответствующем порте. Настройка функции Port Security осуществляется следующим образом:

```
Switch(config)#int range fa0/1 - 20
Switch(config-if-range)#switchport port-security
Switch(config-if-range)#switchport port-security violation protect
Switch(config-if-range)#switchport port-security maximum 1
Switch(config-if-range)#switchport port-security mac-address sticky
```

Первой командой мы задаем диапазон портов (все пользовательские порты), второй - включаем Port Security. Третья строка (**violation**) определяет режим безопасности, в нашем случае это **Protect**. Затем мы определяем максимальное количество MAC-адресов на порту (в нашем примере это 1 MAC). В последней строке параметр **sticky** указывает на то, что нужно запомнить текущий MAC-адрес и не пытаться его снова изучать после перезагрузки. При этом мы можем вручную привязывать к порту статический MAC-адрес. Делается это в режиме настройки конкретного интерфейса:

```
Switch(config)#int fa0/1
Switch(config-if)#switchport port-security mac-address abcd.xxxx.1234
```

Однако на практике чаще всего используют именно **sticky**, т.е. к порту автоматически привязывается MAC-адрес, который коммутатор “видит” в момент настройки. Теперь чуть подробнее о режимах безопасности. Режим **protect** - это самая мягкая реакция на превышении допустимого количества MAC-адресов. В случае превышения лимита (или появления “чужого” MAC-адреса) трафик будет просто отбрасываться. Режим **restrict** аналогичный, но в дополнение отправляется SNMP-трап и делается запись в syslog об инциденте. Самый “жесткий” режим - **shutdown**. Как следует из названия, реакция при нарушении политики - отключение порта (переход в состояние **error-disabled**). Включить данный порт можно будет либо вручную (ввести сначала команду **shutdown**, а затем **no shutdown**), либо настроить автоматическое включение через заданное время (команда **errdisable recovery cause psecure-violation**).

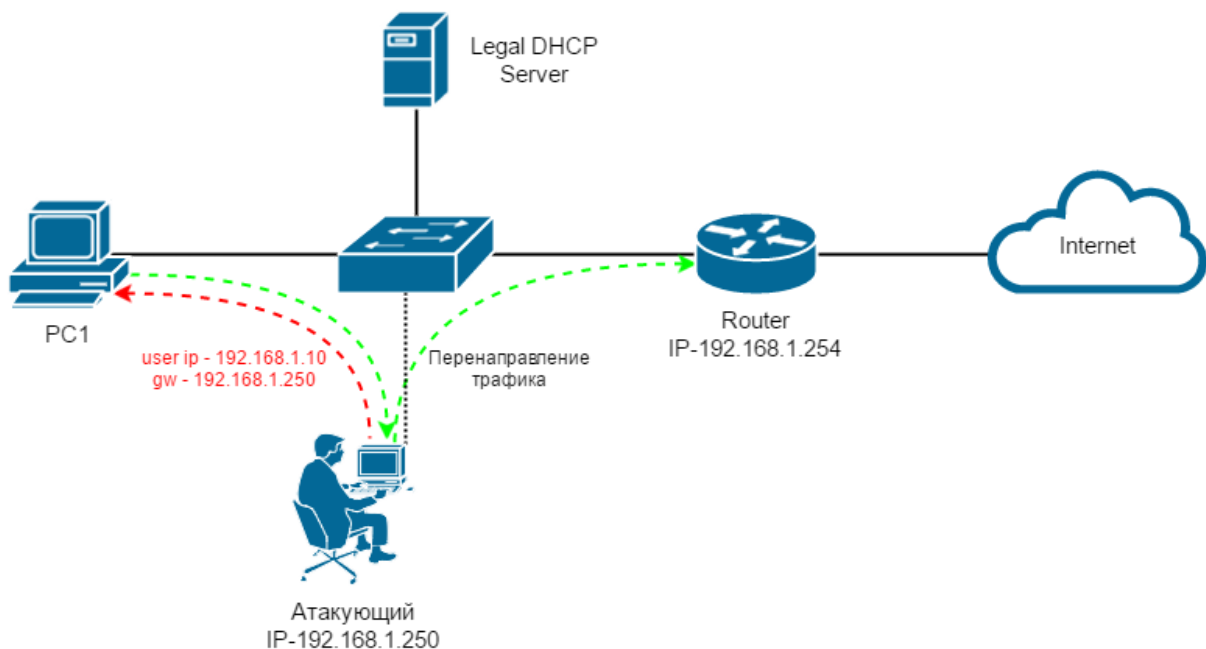
При построении защищенной локальной сети, функция Port Security одна из самых важных. Не пренебрегайте данным инструментом.

3.1.11 DHCP snooping

Невозможно представить локальную сеть без использования **DHCP-сервера**. Это один из самых важных сервисов и соответственно любая проблема с DHCP моментально отражается на работоспособности всей сети. Несмотря на всю важность этого сервиса, многие администраторы забывают про необходимость его защиты. А нарушить работу DHCP довольно просто, причем сделать это может не только

обученный хакер, но и самый обычный пользователя, сам того не желая. Одна из самых больших угроз для DHCP - появление второго “нелегального” DHCP сервера. Могу поведать свою историю, когда я в начале своей карьеры случайно “положил” сеть всей компании. Собирая очередной макет в GNS3, я воспользовался VMware Workstation и “прокинул” один из VMnet-ов в физическую сеть. При этом я совершенно забыл (а может и не знал в то время), что по умолчанию на VMnet-е включен DHCP сервер. В итоге в сети компании появился второй DHCP-сервер (по факту мой компьютер), на который начали “подсаживаться” пользователи, получая неправильный IP-адрес и шлюз по умолчанию. Это парализовало сеть на целый час, пока мы выясняли что случилось. К такому же результату может привести подключенный домашний роутер пользователя, на котором по “дефолту” включен DHCP-сервер.

При этом если неосторожные действия пользователя могут привести лишь к простую сети, то для опытного хакера подобная уязвимость DHCP превращается в отличную возможность перехвата трафика. Для этого ему достаточно организовать DHCP-сервер и стать шлюзом по умолчанию для пользователей. Все запросы он будет маршрутизировать дальше, имитируя нормальную работу сети. Однако с этого момента он будет видеть весь трафик и может его модифицировать (атака man in the middle).



Для избежания подобных проблем существует **DHCP snooping**. Данная функция позволяет определить за каким портом разрешен DHCP-сервер. Т.е. порт будет являться доверенным (**trust**) и на нем будут транслироваться все DHCP ответы. Другие же порты помечаются как недоверенные (**untrust**) и все DHCP-ответы будут отбрасываться, что делает невозможным работу “нелегального” DHCP-сервера. Вернемся к картинке выше и предположим, что легальный DHCP-сервер подключен к порту FastEthernet 0/15. В таком случае настройка DHCP Snooping будет выглядеть следующим образом:

```
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp-snooping vlan 2
Switch(config)#int fa0/15
Switch(config-if)#ip dhcp snooping trust
```

Первой командой мы глобально включаем dhcp snooping. Затем определяем для каких VLAN-ов мы включаем защиту (если их несколько, то нужно указать для каждого). Затем заходим в настройки порта Fa0/15 и объявляем его доверенным (trust). С этого момента все остальные порты являются недоверенными, а сервис DHCP сможет функционировать только на порту FastEthernet 0/15.

Обязательно используйте DHCP Snooping для защиты DHCP-сервиса.

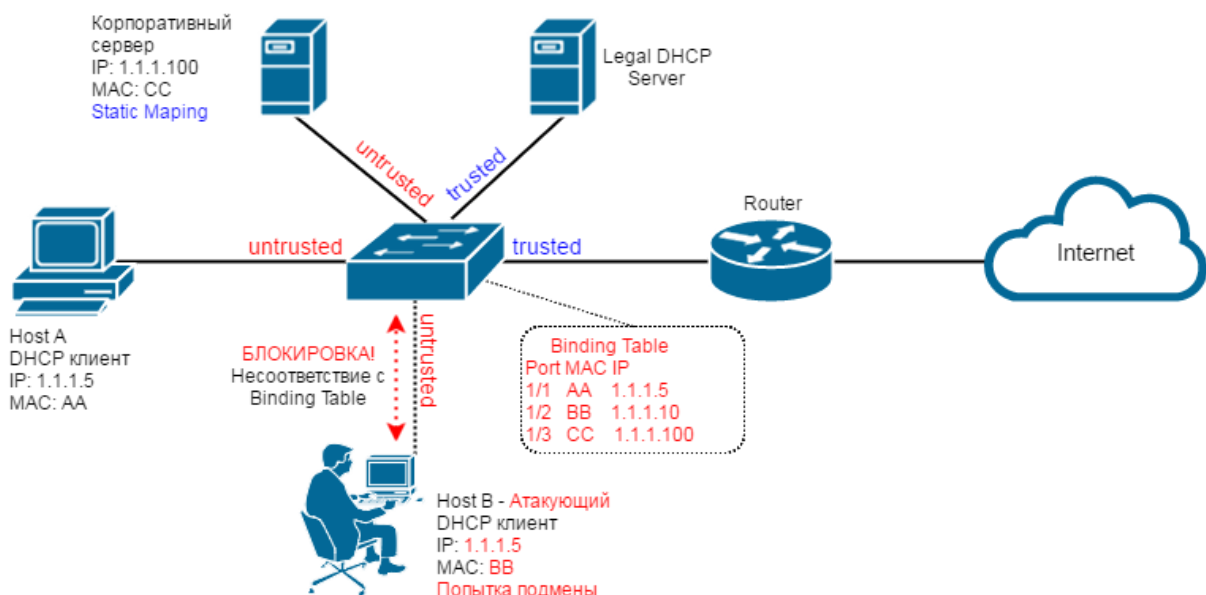
Ни в коем случае не пренебрегайте данной рекомендацией, т.к. описанные угрозы рано или поздно обязательно проявляются в каждой корпоративной сети.

3.1.12 IP Source Guard

При включенном DHCP Snooping коммутатор автоматически создает базу соответствия портов, MAC- и IP-адресов. База формируется за счет записи всех DHCP запросов и ответов. С такой таблицей соответствия (Binding Table) появляется возможность борьбы с атаками типа **IP Spoofing** (подмена IP-адреса). Для этого существует функция **IP Source Guard**, которая проверяет каждый пакет на:

- соответствие IP-адреса источника с адресом, который был ранее получен из базы DHCP Snooping;
- соответствие MAC-адреса источника с адресом из той же базы DHCP Snooping.

Т.е. если пользователь Host A подключился к порту FastEthernet 1/1 и получил IP-адрес от DHCP-сервера, то в таблицу добавляется запись об этом порте, MAC- и IP-адресе. Если после этого, кто-то попытается представиться как Host A (подменив IP или MAC), то у него ничего не выйдет, т.к. в таблице уже есть соответствующая запись.



Включается IP Source Guard командой для конкретного интерфейса:

```
Switch(config)#int fa1/1  
Switch(config-if)#ip verify source vlan dhcp-snooping port-security
```

Данная функция используется на всех недоверенных портах (как правило это все пользовательские порты). Следует отметить, что для работы IP Source Guard

требуется включенный DHCP snooping, а для контроля MAC-адресов должен быть включен Port Security. Для серверов можно использовать статическую привязку:

```
Switch(config)#ip source binding mac-address vlan vlan-id ip-address interface interface-name
```

Активируйте IP Source Guard для защиты от IP Spoofing-a.

3.1.13 Dynamic ARP Inspection

Наверняка всем знаком принцип работы протокола **ARP**: посылается широковещательный запрос вида “у кого ip-адрес X.X.X.X?”, после чего, устройство с IP-адресом X.X.X.X отвечает. Подобная схема уязвима для атаки типа **ARP-poisoning** (или **ARP-spoofing**). Во время атаки вместо настоящего хоста с адресом X.X.X.X отвечает хост злоумышленника, который в итоге перехватывает трафик. Для предотвращения подобных атак используется функция **Dynamic ARP Inspection (DAI)**. Алгоритм работы похожа на DHCP-Snooping: порты делятся на доверенные и недоверенные. На недоверенных каждый ARP-ответ подвергаются анализу: сверяется информация, содержащаяся в этом пакете, с той, которой коммутатор доверяет (либо статически заданные соответствия MAC-IP, либо информация из базы DHCP Snooping). Если не сходится - пакет отбрасывается и генерируется сообщение в syslog.

Чтобы активировать DAI в нужном VLAN-е используйте следующую команду:

```
Switch(config)#ip arp inspection vlan 2
```

По умолчанию все порты недоверенные. Чтобы сделать порт доверенным:

```
Switch(config-if)#ip arp inspection trust
```

Настройте Dynamic ARP Inspection для защиты от ARP Spoofing-a.

3.1.14 IEEE 802.1X

Говоря о защите уровня доступа невозможно не вспомнить такую технологию как **IEEE 802.1X**. И вот почему.

Конвергентный и адаптивный доступ к сети становится все более популярным в последние годы. Во многих компаниях уже нет в принципе стационарных рабочих мест и организация доступа на основе IP- или MAC-адреса становится весьма затруднительной. Пользователь может несколько раз за день поменять свою локацию, при этом он должен находиться в строго определенном VLAN-е, согласно его правам доступа. Как правило, для решения данной задачи используется IEEE 802.1X.

IEEE 802.1X (dot1X) — технология предназначенная для аутентификации и авторизации на канальном уровне. Пользователь, подключившись к сетевой розетке, не сможет передавать никакой трафик (кроме сообщений 802.1X), пока не пройдет процедуру аутентификации. Авторизовавшись, пользователь может автоматически

помещаться в тот VLAN, который определен политиками безопасности, независимо от точки подключения. К этому VLAN можно привязать списки контроля доступа, политики QoS и т.д., таким образом полностью персонализировать сетевой доступ.

Собрать стенд для тестирования 802.1X довольно просто. Для начала вам понадобится **RADIUS-сервер**. Он будет обрабатывать запросы на аутентификацию, поступающие от коммутатора с поддержкой 802.1X. Также необходим клиент 802.1X на рабочей станции, который сможет передать необходимые параметры коммутатору. В этой схеме свитч выступает как посредник между RADIUS-сервером и рабочей станцией в процессе аутентификации, а также обеспечивает контроль доступа к порту и непосредственно авторизацию.

Однако стоит заметить, что при практическом внедрении 802.1X в корпоративную сеть придется учесть множество моментов: выбор программного обеспечения для аутентификации, настройка протокола IP на рабочих станциях, различные сценарии работы пользователей в сети, согласование групповых политик и так далее. Данная тема настолько обширна, что выходит за рамки нашей книги. Приведенная выше информация предназначена лишь для базового ознакомления с этой технологией.

3.2 Уровень распределения и ядра

После уровня доступа следует уровень распределения. Однако, очень часто он объединяется с уровнем ядра - так называемый collapsed core. При этом, в качестве устройств уровня распределения/ядра могут выступать как L3, так и L2 коммутаторы. Все зависит от размеров и архитектуры сети. Ключевая задача этих устройств - агрегация и распределение трафика. Функции защиты и безопасности в данном случае уходят на второй план. Однако, кроме обеспечения защищенного доступа, есть еще несколько аспектов, которые мы обсудим ниже.

3.2.1 STP priority

Настраивая коммутатор распределения/ядра вы должны в первую очередь убедиться, что он будет являться корневым для протокола STP. Это особенно важно в отказоустойчивой сети с избыточной топологией (наличие резервных линков). Как мы уже говорили выше, корневым становится коммутатор с наименьшим **Bridge Priority**. По умолчанию данный параметр равен **32768**. Однако, чтобы не быть привязанным к конкретному числу, в коммутаторах Cisco есть возможность вручную указать какой коммутатор будет всегда являться корневым (bridge priority автоматически пересчитывается в случае обнаружения более низкого значения). Для этого используется команда типа:

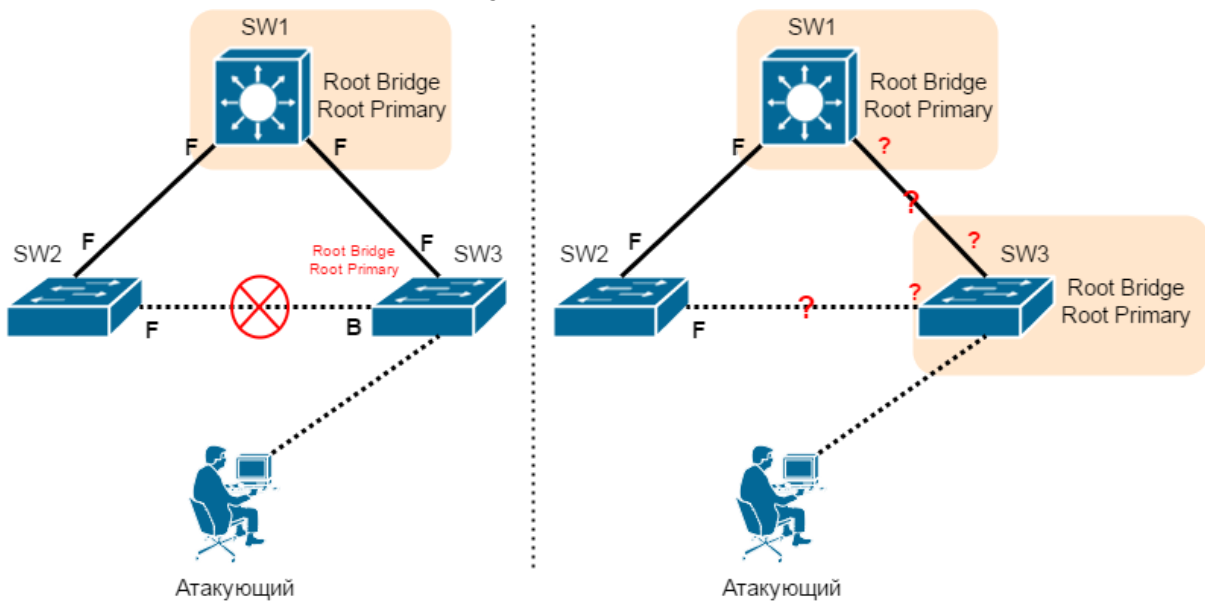
```
Switch1(config)#spanning-tree vlan 2,3 root primary
```

Таким образом Switch1 будет являться корневым для VLAN 2 и 3. Это избавит вас от “неправильного” срабатывания STP, когда трафик начинает идти не самым оптимальным путем. Используйте подобную команду для каждого создаваемого влана на коммутаторах уровня распределения/ядра.

Коммутатор распределения/ядра должен быть корневым для всех VLAN-ов.

3.2.2 Root Guard

Вернемся к вопросу, который мы рассматривали в пункте 3.1.9. Как вы помните, мы описывали уязвимость, связанную с рассылкой BPDU пакетов с компьютера злоумышленника, что могло привести к перестроению STP топологии и смене корневого коммутатора. На уровне доступа мы решили данную проблему с помощью **BPDU Guard** на всех недоверенных (пользовательских) портах. Однако, если представить, что злоумышленнику удалось “взломать” коммутатор доступа и изменить **bridge priority**. Это приведет либо к смене корневого коммутатора, либо к коллизии, т.к. в сети появятся два **root primary**.



Чтобы избежать этого, на коммутаторах распределения/ядра необходимо активировать дополнительную функцию - **Root Guard**. Включается данная защита на конкретном порте, где мы в принципе запрещаем появление корневого коммутатора. В нашем случае это линки до коммутаторов доступа. Пример:

```
Switch1(config)#interface range Gi0/1-2  
Switch1(config-if-range)# spanning-tree guard root
```

Это позволит блокировать любые попытки смены корневого коммутатора. Будьте осторожны с данной функцией, т.к. неправильная настройка может привести к проблемам с “петлями”, особенно в сложных топологиях с несколькими избыточными линками.

Используйте **Root Guard** на коммутаторах распределения/ядра.

3.2.3 RSTP

Данную тему можно было затронуть и раньше, но я сознательно перешел к ней лишь в параграфе про уровень распределения. Попробую объяснить почему.

Ранее мы обозначили, что STP это протокол защиты от “петель”. По умолчанию он включен на всех коммутаторах. Однако, вероятность образования “петли” в сети с одним коммутатором - крайне мала. Про STP вспоминают только когда появляются резервные линки (избыточная топология). И чаще всего это бывает в больших и средних сетях, где есть уровень распределения или ядра. Именно поэтому я решил обсудить этот вопрос только сейчас, а не раньше.

Протокол STP является довольно старой разработкой. Появился он еще на заре возникновения сетей (в 1985 году). Один из самых главных недостатков данного протокола - скорость. Время перестроения топологии (в случае обнаружения петли или обрыва основного линка) может достигать до 30-50 секунд, что совершенно неприемлемо в наши дни. Именно поэтому появилась более новая версия - **RSTP**. Возможно данная тема больше касается архитектурных особенностей построения сетей. Однако простой в сети тоже является угрозой безопасности.

RSTP (Rapid spanning tree protocol или быстрый протокол разворачивающегося дерева) - улучшенная версия протокола STP. Главное достоинство - высокая скорость перестроения топологии. Время сходимости в данном случае меньше 10 секунд. В небольших сетях момент переключения линков практически незаметен.

Мы не будем подробно рассматривать работу данного [протокола](#) и ограничимся описанием настройки:

```
Switch(config)#spanning-tree mode rapid-pvst
```

*Пользуйтесь **RSTP** вместо более медленного **STP**.*

3.2.4 Общие рекомендации

Как уже говорилось ранее, функции защиты не являются приоритетными для коммутаторов распределения/ядра. Но это не значит, что на данном уровне можно забыть о безопасности. В первую очередь вы должны обеспечить безопасный доступ к оборудованию (как физический, так и удаленный). Затем воспользоваться большинством ранее приведенных рекомендаций. Однако стоит понимать, что в отличие от уровня доступа, сеть уровня распределения/ядра как правило статична. К таким коммутаторам не подключаются пользователи, а физический доступ почти всегда ограничен (серверной комнатой или шкафом). В связи с этим, некоторые функции защиты могут быть излишни, в зависимости от вашей архитектуры. Какие именно угрозы актуальны в вашей сети? Хорошо обдумайте этот вопрос прежде чем приступить к настройке уровня распределения/ядра.

Следует заметить, что для уровня распределения/ядра также актуальны вопросы формирования списков доступа (access-list). Данный вопрос мы рассмотрим в следующей главе, дабы не растягивать текущую.

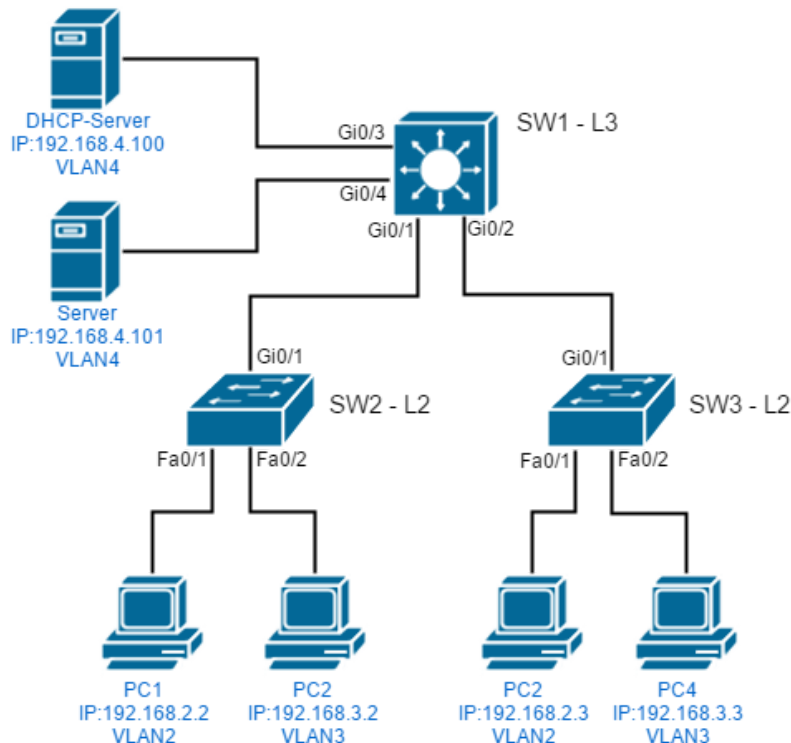
3.3 Чеклист №3

Третья глава подошла к концу, резюмируем все вышесказанное в виде Чек-листа №3:

1. Избегайте “больших” широковещательных доменов. Сегментируйте сеть для создания структуры и порядка.
2. Отключайте неиспользуемые порты, либо переводите их в неиспользуемый VLAN.
3. Не используйте VLAN 1.
4. Все порты должны иметь строго определенный режим: **mode access** или **mode trunk**.
5. Не разрешайте все VLAN-ы в Trunk - портах. С помощью команды **switchport trunk allowed vlan** указывайте конкретные VLAN-ы.
6. Указывайте **description** для ключевых портов коммутатора и давайте имена всем создаваемым VLAN-м.
7. При необходимости настройте автоматическую защиту от широковещательных штормов.
8. Активируйте функцию **PortFast** на портах пользователей.
9. Используйте **BPDU Guard** на пользовательских портах.
10. Используйте **Port Security** на всех пользовательских портах.
11. Обязательно используйте **DHCP Snooping** для защиты DHCP-сервиса.
12. Активируйте **IP Source Guard** для защиты от **IP Spoofing**-а.
13. Настройте **Dynamic ARP Inspection** для защиты от **ARP Spoofing**-а.
14. Коммутатор распределения/ядра должен быть корневым для всех VLAN-ов.
15. Используйте **Root Guard** на коммутаторах распределения/ядра.
16. Пользуйтесь **RSTP** вместо более медленного **STP**.

3.4 Пример конфигурации

В качестве примера рассмотрим типовую (сильно упрощенную) схему локальной сети, не затрагивая оборудование выхода в интернет (Internet Edge):



Пример настроек касается только тем, которые были затронуты в данной главе.

1) Настройки SW2

Отключаем неиспользуемые порты:

```
Switch(config)#interface range fa0/3 - 24
Switch(config-if)#shutdown
```

Создаем VLAN-ы:

```
Switch(config)#vlan 2
Switch(config-vlan)#name UsersA
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#name UsersB
Switch(config-vlan)#exit
```

Настраиваем порт Fa0/1:

```
Switch(config)#int fa0/1
Switch(config-if)#description UserA1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#storm-control broadcast level 10.00
Switch(config-if)#storm-control action shutdown
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security violation protect
Switch(config-if)#switchport port-security maximum 1
Switch(config-if)#switchport port-security mac-address sticky
Switch(config-if)#ip verify source vlan dhcp-snooping port-security
```

Настраиваем порт Fa0/2:

```
Switch(config)#int fa0/2
Switch(config-if)#description UserB1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#storm-control broadcast level 10.00
Switch(config-if)#storm-control action shutdown
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security violation protect
Switch(config-if)#switchport port-security maximum 1
Switch(config-if)#switchport port-security mac-address sticky
Switch(config-if)#ip verify source vlan dhcp-snooping port-security
```

Настраиваем порт Gi0/1:

```
Switch(config)#int gi0/1
Switch(config-if)#description Link-to-SW1
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 2,3
Switch(config-if)#ip dhcp snooping trust
```

Настраиваем глобальные параметры:

```
Switch(config)#errdisable recovery cause storm-control
Switch(config)#errdisable recovery interval 300
Switch(config)#spanning-tree portfast default
Switch(config)#spanning-tree portfast bpduguard default
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp-snooping vlan 2,3
Switch(config)#ip arp inspection vlan 2,3
Switch(config)#spanning-tree mode rapid-pvst
```

2) Настройки SW3

Настройки коммутатора SW3 будут абсолютно аналогичны настройкам SW2. Единственное отличие это в названии портов. Логичнее дать имена UserA2 и UserB2.

3) Настройки SW1

Отключаем неиспользуемые порты:

```
Switch(config)#interface range gi0/5 - 24
Switch(config-if)#shutdown
```

Создаем VLAN-ы:

```
Switch(config)#vlan 2
Switch(config-vlan)#name UsersA
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#name UsersB
Switch(config-vlan)#exit
Switch(config)#vlan 4
Switch(config-vlan)#name SRV
Switch(config-vlan)#exit
```

Настраиваем порты:

```
Switch(config)#int gi0/1
Switch(config-if)#description Link-to-SW2
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 2,3
Switch(config-if)#spanning-tree guard root
Switch(config-if)#exit
```

```
Switch(config)#int gi0/2
Switch(config-if)#description Link-to-SW3
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 2,3
Switch(config-if)#spanning-tree guard root
Switch(config-if)#exit
```

```
Switch(config)#int gi0/3
Switch(config-if)#description DHCP-SRV
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 4
Switch(config-if)#ip dhcp snooping trust
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security violation protect
Switch(config-if)#switchport port-security maximum 1
Switch(config-if)#switchport port-security mac-address abcd.xxxx.1234
Switch(config-if)#exit
```

```
Switch(config)#int gi0/4
Switch(config-if)#description Server
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 4
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security violation protect
Switch(config-if)#switchport port-security maximum 1
Switch(config-if)#switchport port-security mac-address abcd.xxxx.5678
Switch(config-if)#ip verify source vlan dhcp-snooping port-security
Switch(config-if)#exit
```

Настраиваем глобальные параметры:

```
Switch(config)#spanning-tree portfast default
Switch(config)#spanning-tree portfast bpduguard default
Switch(config)#spanning-tree vlan 2,3,4 root primary
Switch(config)#spanning-tree mode rapid-pvst
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp-snooping vlan 2-4
```

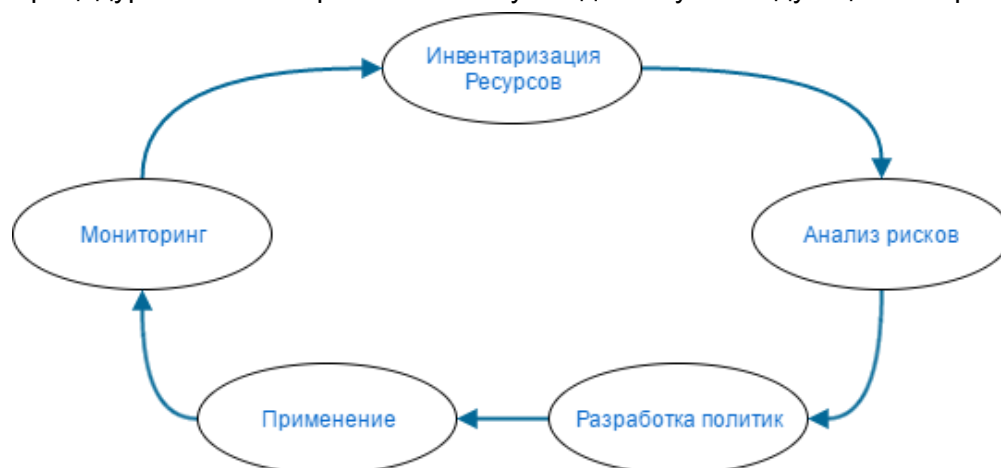
Switch(config)#ip arp inspection vlan 2-4

Стоит отметить, что поддержка тех или иных команд сильно зависит от модели коммутатора и версии его прошивки.

4. Управление безопасностью

Мы добрались до тем, которые старательно избегает практически любой системный администратор или сетевой инженер. Управление безопасностью, Модель угроз, Анализ рисков, Политика безопасности и т.д. Со всем этим рано или поздно приходится сталкиваться любому специалисту, который занимается вопросами Информационной безопасности.

Процесс построения защищенной системы весьма непростая, а порой ужасно нудная процедура. В самом простейшем случае действует следующий алгоритм:



Т.е. как можно видеть из рисунка, информационная безопасность это непрерывный процесс. Пожалуй это самое главное, что вы должны запомнить после прочтения этой главы.

Информационная безопасность это не результат, а непрерывный процесс.

Стоит отметить, что в рамках данной книги мы не сможем подробно рассмотреть все пункты. Автор не ставил перед собой задачи сделать из вас эксперта по информационной безопасности (которым он сам и не является). Вместо этого мы затронем весьма узкую область, а именно формирование списков доступа. Это позволит гораздо быстрее начать процесс построения защищенной сети, не вникая в академические труды. Многим может показаться такой подход “халтурой”, но на мой взгляд такой подход гораздо эффективнее, особенно в том случае, когда специалисту приходится совмещать на работе несколько ролей. “Лучше уж так, чем вообще никак”. Я постараюсь описать самые основные принципы, которых следует придерживаться.

При этом стоит понимать, что для построения действительно серьезной и комплексной защиты, вы будете просто обязаны пройти специализированные курсы, прочитать соответствующие руководства, либо обратиться за помощью к квалифицированным людям.

P.S. Мне бы хотелось перейти сразу к теме защиты периметра, однако мы не можем этого сделать без небольшого экскурса в терминологию информационной безопасности. Следующие несколько параграфов будут “слегка” выбиваться из общего стиля изложения, однако я должен осветить некоторые моменты.

4.1 Иерархия средств защиты

Вопреки расхожему мнению, средства защиты это не только межсетевые экраны и системы предотвращения вторжений (IPS). Все средства защиты можно разделить на четыре иерархически организованных уровня. Средства каждого уровня могут быть использованы на разных этапах жизненного цикла системы обеспечения информационной безопасности (данный параграф приводится не только для расширения кругозора, но и для осознания, насколько сложен процесс построения комплексной защищенной системы).

- **Законодательный уровень.** К нему относится правовое регулирование, стандартизация, лицензирование и морально-этические нормы, принятые в обществе. Законодательство может прямо влиять на концепцию построения защиты. Например, выход Закона РФ “О персональных данных”, потребовал от многих компаний пересмотра собственной информационной безопасности. Тут же можно вспомнить “Закон Яровой”, который явно отражается на многих ИТ-инфраструктурах.
- **Административный уровень.** Основу административного уровня составляет политика безопасности, которая определяет стратегические направления информационной защиты компании. На данном уровне очерчивается круг критически важных информационных ресурсов, защита которых представляет наивысший приоритет. Также предлагаются меры по снижению рисков. На основе полученной стратегии разрабатывается программа обеспечения безопасности, планируется бюджет, назначаются руководители и очерчивается их зона ответственности. В следующем параграфе мы вкратце рассмотрим процесс создания политики безопасности.
- **Процедурный уровень.** Этот уровень является промежуточным между Административным и Техническим. В качестве средства Процедурного уровня выступает человек, который выполняет определенные действия для решения проблем с информационной безопасностью. Любой аспект защиты требует процедурного уровня. Например, в каждой организации должно осуществляться резервное копирование, смена паролей, профилактические работы и т.д. Процедурный уровень описывает эти повседневные или периодические задачи. Таким образом Информационная безопасность становится непрерывным процессом.
- **Технический уровень.** К этому уровню относятся **программные, аппаратные и программно-аппаратные** средства защиты. Программные средства включают защитные инструменты операционных систем (подсистемы аутентификации и авторизации, средства управления доступом, аудит и т.д.) и прикладные программы обеспечения безопасности (антивирусные средства, прокси-сервера, системы предотвращения вторжений, программные межсетевые экраны и т.д.). Примером аппаратных средств являются источники бесперебойного питания, генераторы напряжения, средства контроля доступа в помещение и т.д. К аппаратно-программным средствам относятся межсетевые экраны, маршрутизаторы, сетевые анализаторы. И хотя данный уровень

называется техническим, к нему также относятся математические методы (криптография), алгоритмы, абстрактные модели и т.д.

Не ограничивайтесь Техническим уровнем защиты.

Стоит отметить, что большая часть книги посвящена именно Техническому уровню, но с некоторыми ограничениями. В самом начале книги мы обусловились, что не обладаем специализированными средствами защиты и будем рассматривать возможности коммутационного и маршрутизирующего оборудования с точки зрения информационной безопасности. Если же вас интересуют программно-аппаратные средства, то рекомендую к ознакомлению книгу [“Архитектура корпоративных сетей”](#).

4.2 Политика безопасности

Как было сказано в самом начале книги, при построении информационной защиты нельзя полностью полагаться на технические средства. Даже самые современные межсетевые экраны или системы предотвращения вторжений не защитят организацию, если нет конкретной цели и хотя бы примерного плана по ее достижению. Должна быть выбрана руководящая идея, а для этого мы должны располагать хотя бы общими соображениями, в каком направлении двигаться. Таким образом мы автоматически приходим к понятию **“Политика безопасности”**.

Информационная безопасность компании невозможна без утвержденной Политики безопасности.

В общем смысле **“Политика”** - это руководство, которое устанавливает главные направления, в которых нужно действовать, чтобы наиболее рациональным путем достичь поставленной цели. Как правило, Политика безопасности представляет собой три уровня:

- 1) **Верхний уровень** - решения, затрагивающие компанию в целом, они принимаются высшим руководством (генеральный директор, начальник и т.д.), носят общий характер.
- 2) **Средний уровень** - решения, относящиеся к отдельным аспектам обеспечения информационной безопасности (политика доступа в Интернет, парольная политика, антивирусная политика и т.д.).
- 3) **Нижний уровень** - решения, касающиеся регламентации отдельных сервисов (правила пользования электронной почтой, ограничение доступа к файлам и т.д.).

Как правило, каждый уровень представляется в виде отдельного набора документов. Чем ниже уровень, тем более детальное описание. В небольших компаниях все уровни могут быть представлены в рамках одного документа.

Разработка подобного перечня документов может показаться “кошмаром” для неподготовленного человека. Однако, Политика безопасности является одним из самых важных элементов при создании защищенной ИТ-инфраструктуры. Ниже мы подробнее рассмотрим каждый уровень.

4.2.1 Верхний уровень

Первое с чего стоит начать, так это с принятия решения о том, что компания действительно нуждается в системе обеспечения информационной безопасности. Данное решение может быть принято только руководством компании. Причины могут быть разные: произошедший инцидент ИБ (шифровальщики, кража данных), соблюдение законодательства (Закон о персональных данных), требования различных регуляторов и т.д. Должны быть весомые аргументы, которые обосновывают необходимость в защите.

Для системного администратора или инженера данный документ очень важен, т.к. при построении защищенной сети ему наверняка придется сталкиваться с возражениями пользователей, которые не хотят менять пароль несколько раз в год, использовать смарт-карту для входа в систему или запрашивать доступ к файловому хранилищу. В этом случае всегда можно показать Политику безопасности компании, которая подписана директором и является подобием “закона”, который все должны соблюдать.

Затем определяются границы разрабатываемой Политики безопасности (далее ПБ):

- Все ли сегменты сети должны быть включены в ПБ?
- Всех ли сотрудников касается ПБ?
- Учитывать ли в ПБ удаленных сотрудников?
- Все ли сервисы компании должны быть включены в ПБ?
- И т.д.

После этого выполняется обследование компании: выявляются критически важные сервисы/активы, устанавливаются правила разграничения доступа к информационным ресурсам, определяются наиболее вероятные угрозы, оцениваются возможные потери, принимаются концептуальные решения относительно методов обеспечения защиты. Т.е. значительная часть ПБ основывается на результатах анализа рисков.

Адекватность Политики безопасности сильно зависит от правильного Анализа рисков.

Как правило верхний уровень ПБ это документ на 2-5 страниц, который ссылается на какие-либо законы, стандарты, частные политики, регламенты и руководства.

4.2.2 Средний уровень

Этот уровень часто называют уровнем частных политик, т.к. к нему относят решения и соответствующие документы, касающиеся частных аспектов информационной безопасности таких, как парольная политика, антивирусная политика, политика защиты коммуникационных каналов, политика использования криптографических средств и т.д. При этом в ПБ верхнего уровня должны быть ссылки на абсолютно все частные политики, дабы понимать с какой целью они используются.

В политике верхнего уровня должны быть ссылки на все частные.

В качестве примера можно привести частную политику, которая касается подключения к корпоративной сети персональных устройств пользователей. В рамках политики мы можем обозначить, что:

- персональные устройства могут подключаться к гостевой сети;
- запрещено бесконтрольное подключение персональных устройств к корпоративной сети;
- подключение персональных устройств к корпоративной сети возможно в случае получения разрешения у специалиста ИБ;
- и т.д.

Документы, описывающие частные политики, не имеют жесткого формата, их содержание зависит от специфики компании, от того, какой аспект ИБ они затрагивают. Частные политики иногда играют роль корпоративных стандартов и могут являться конфиденциальными.

4.2.3 Нижний уровень

Политики нижнего уровня определяют действия по обеспечению безопасности на уровне сетевых сервисов и могут представлять собой руководства, инструкции, регламенты и правила, связанные с администрированием и использованием сервисов.

Выше мы рассмотрели несколько пунктов из частной политики безопасности по использованию персональных устройств в корпоративной сети. Однако что является персональным устройством? Чем отличается гостевая сеть от корпоративной? Как получить разрешение на использование? На эти и многие другие вопросы отвечает ПБ нижнего уровня.

Политика безопасности должна быть понятна обычным пользователям, не являющихся техническими специалистами.

При этом не стоит путать обычные инструкции с ПБ. Политика описывает процесс использования баз данных, электронной почты, файлового хранилища, антивирусных программ и т.д. Т.е. касается сервисов, которые оказывают сильное влияние на информационную безопасность компании (именно поэтому ПБ согласовывается с руководством). В некоторых случаях ПБ нижнего уровня могут отсутствовать, ввиду достаточности и понятности Политик среднего. Инструкции пользователей должны присутствовать в обязательном порядке.

4.2.4 Пример политики безопасности

Изначально мне хотелось привести хотя бы краткий пример Политики безопасности, однако это будет слишком объемная часть и сильно выбиваться из формата книги. Поэтому было принято решение, оставить данную тему на самостоятельное изучение. Для этого в поиске (google) можно использовать запрос типа “пример политики информационной безопасности”. Вы найдете огромное количество примеров, которые останется лишь адаптировать под себя.

Стоит отметить, что создать адекватную ПБ с первого раза весьма трудно. Хорошей практикой является периодический пересмотр всей Политики на предмет возможного улучшения и оптимизации. Вы можете изначально придумать очень строгую ПБ, однако вскоре окажется, что эффективность работы пользователей сильно упала из-за огромного количества неудобств, связанных с соблюдением вашей Политики. Ищите баланс.

*Безопасность это всегда компромисс между защищенностью и удобством
пользователей.*

На самом деле, было бы гораздо логичнее начать книгу именно с 4-ой главы, поскольку защита сети должна начинаться как раз с утвержденной ПБ. Уже на основе выбранной Сетевой политики безопасности можно создавать списки доступа и использовать функции защиты описанные в первых главах. Однако формат данной книги предполагает экспресс погружение, поэтому порядок “слегка” изменен. Далее мы более подробно рассмотрим вопрос формирования списков доступа.

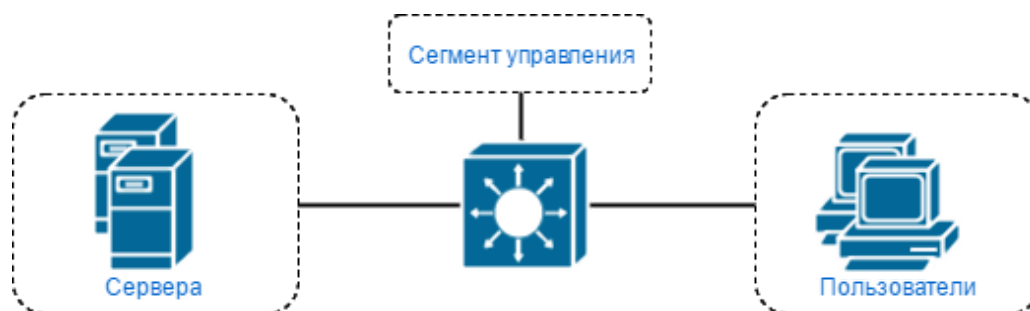
4.4 Чеклист №4

- 1) *Информационная безопасность это **не результат**, а **непрерывный процесс**.*
- 2) *Не ограничивайтесь Техническим уровнем защиты.*
- 3) *Информационная безопасность компании невозможна без утвержденной Политики безопасности.*
- 4) *Адекватность Политики безопасности сильно зависит от правильного Анализа рисков.*
- 5) *В политике верхнего уровня должны быть ссылки на все частные.*
- 6) *Политика безопасности должна быть понятна обычным пользователям, не являющихся техническими специалистами.*
- 7) *Безопасность это всегда компромисс между защищенностью и удобством пользователей.*

5. Управление доступом

В начале третьей главы мы обозначили, что Сегментация - это первое, с чего начинается защищенная сеть. Однако мы рассматривали сегменты с точки зрения упорядочивания сети. В этот раз мы рассмотрим их с позиции безопасности.

Данная тема касается не только защиты периметра, но и защиты локальной сети. Если вспомнить типовую сеть, то можно обнаружить L3 коммутатор на уровне распределения, который как правило используется для сегментирования локальной сети.



На этом коммутаторе часто выделяется сегмент пользователей, локальных серверов, сегмент управления и так далее. Разбивая сеть таким образом вы получаете возможность контролировать все соединения между выделенными сегментами. Однако тут же появляется необходимость в правилах разграничения - списках доступа (или **access-list**). Без этих правил польза от сегментов весьма посредственная (разве что ограничение широковещательных доменов). С точки зрения безопасности, главная цель сегмента - контроль доступа. Это позволяет минимизировать риск от потенциального взлома. Если предположить, что злоумышленник смог получить доступ к компьютеру пользователя, то грамотно сформированные access-list-ы не позволят ему выйти за пределы сегмента и взломать более ценные сервера компании.

Не оставляйте сегменты без соответствующих списков доступа.

Вопрос формирования списков доступа довольно часто ставит администраторов в тупик и как результат - разрешен абсолютно весь трафик между сегментами. Это множит на ноль все усилия потраченные на создание защищенной сети.

В этой главе мы попробуем разобраться каким образом сформировать access-list-ы для защиты локальной сети, т.е. на уровне распределения. Если у вас отсутствует L3 коммутатор, а сегментация локальной сети осуществляется с помощью периметрального роутера (Internet Edge), то данные советы также будут полезны. Вопросы защиты периметра мы рассмотрим в следующей главе.

5.1 С чего начать?

Уверен, что это первый вопрос, который возникает в голове начинающего инженера или системного администратора, когда перед ним встает задача

сформировать списки доступа. Даже имеющаяся политика безопасности не сильно поможет с технической точки зрения, т.к. в ней приводятся общие задачи. Для настройки же оборудования требуется более детальная информация. Поэтому необходимо начинать с “аудита”.

Проведите “аудит” ИТ-инфраструктуры перед формированием списков доступа.

“Аудит” в данном случае слишком громкое слово, т.к. в простейшем случае нам необходимо ответить хотя бы на четыре главных вопроса:

- 1) Что защищать?
- 2) От чего защищать?
- 3) Как защищать?
- 4) Чем защищать?

В следующих параграфах мы более подробно рассмотрим каждый из этих вопросов.

5.2 Что защищать?

На этом этапе логичнее всего провести инвентаризацию ИТ-инфраструктуры. Составить перечень систем, сервисов и определить наиболее критичные из них. Обычно это оформляется в виде таблицы. Простейший пример:

№	Устройство/Сервис	Идентификатор	Описание
Сетевая инфраструктура			
1	Маршрутизатор	R1	Пограничный маршрутизатор компании
2	Коммутатор	SW2	Коммутатор доступа первого этажа
3
Серверная инфраструктура			
10	Контроллер домена	SrvAD	Сервер Active Directory
11	Сервер служб	Srv1	DHCP, DNS, NTP, FTP сервер
12	Email-сервер	SrvEmail	Почтовый сервер компании
13

Составьте перечень наиболее важных/критичных систем и сервисов.

Естественно, что размер таблицы зависит от размеров и сложности ИТ-инфраструктуры. Иногда бывает удобнее составить отдельные таблицы для устройств (сервера, сетевое оборудование) и отдельные таблицы для сервисов (DNS, DHCP, Email, FTP, WWW и т.д.).

5.3 От чего защищать?

Здесь мы должны понять, что представляет опасность для наших приоритетных сервисов. Утечка информации, несанкционированный доступ, потеря данных, кража?

На данном этапе может составляться некая “**Модель угроз**”, которая описывает все возможные опасности. Это очень емкое понятие, в качестве примера можете ознакомиться с документом “**БАЗОВАЯ МОДЕЛЬ УГРОЗ БЕЗОПАСНОСТИ ПЕРСОНАЛЬНЫХ ДАННЫХ**” утвержденная ФСТЭК. Однако наша книга описывает защиту только на сетевом уровне, поэтому далее мы будем раскрывать именно эту тему.

Некоторые угрозы и механизмы защиты мы уже рассмотрели ранее и не будем их снова описывать. С помощью списков доступа мы можем защититься от следующих угроз (в сети Интернет огромное количество информации об этих угрозах, поэтому мы ограничимся кратким описанием):

- Анализ сетевого трафика (sniffing) - перехват данных с последующим анализом. Особенно актуально если в сети используются не защищенные протоколы (FTP, Telnet, SMTP и т.д.). С помощью списков доступа мы можем заблокировать все уязвимые протоколы, что сделает невозможным их использование (естественно, что это можно делать только при наличии защищенного аналога - SSH, SFTP, SMTPS и т.д.).
- Сканирование сети - определение сетевой топологии, операционных систем, открытых портов, наличие тех или иных сервисов. Данная “разведка” является одной из самых опасных угроз, т.к. получив всю необходимую информацию злоумышленник сможет подготовиться к атаке. Если с помощью списков доступа закрыть все “лишние” порты, то можно существенно усложнить процесс сканирования сети.
- Несанкционированный доступ - название говорит само за себя. С помощью access-list-ов мы можем совершенно конкретно определить кто, как и когда может получить доступ к защищаемому объекту.
- Подмена доверенного объекта сети - spoofing. Некоторые способы борьбы с ним мы уже обсудили (DHCP snooping, IP snooping, ARP snooping). “Правильные” списки доступа будут хорошим дополнением.
- Отказ в обслуживании (DoS) - атака нацелена не на взлом, а на “обрушение”, когда сетью или сервисом становится невозможно пользоваться. Списки доступа здесь также будут полезны.

Кроме того, на данном этапе стоит определить не только “от чего” защищаться, но и “от кого”. От внешних или от внутренних пользователей? Всех пользователей или только от определенной группы? О том, как сделать этот выбор мы поговорим чуть позже, в отдельном параграфе.

Определите от чего и от кого вы будете защищаться.

5.4 Как защищать?

На данном этапе мы определяем, как защититься от возможных угроз. Если рассматривать общую Модель угроз, то здесь можно было бы перечислить уже описанные в первых главах способы. Однако мы рассмотрим исключительно списки доступа. При выборе стратегии защиты (с помощью access-list-ов) есть два основных метода:

- 1) **Разрешено все, что не запрещено.** Это самый часто используемый метод, однако самый небезопасный. Фактически, для маршрутизатора это настройки по умолчанию, когда весь трафик разрешен, пока мы в явном виде что-то не запретили. В некоторых случаях этот режим может быть оправдан, но чаще всего это является результатом халатности и лени администратора.
- 2) **Запрещено все, что не разрешено.** Противоположный подход. По умолчанию весь трафик запрещен, пока в явном виде не будет создано разрешающее правило. Всегда стремитесь использовать именно этот метод, хотя он и требует больше времени на настройку.

При создании списков доступа старайтесь придерживаться метода “Запрещено все, что не разрешено”.

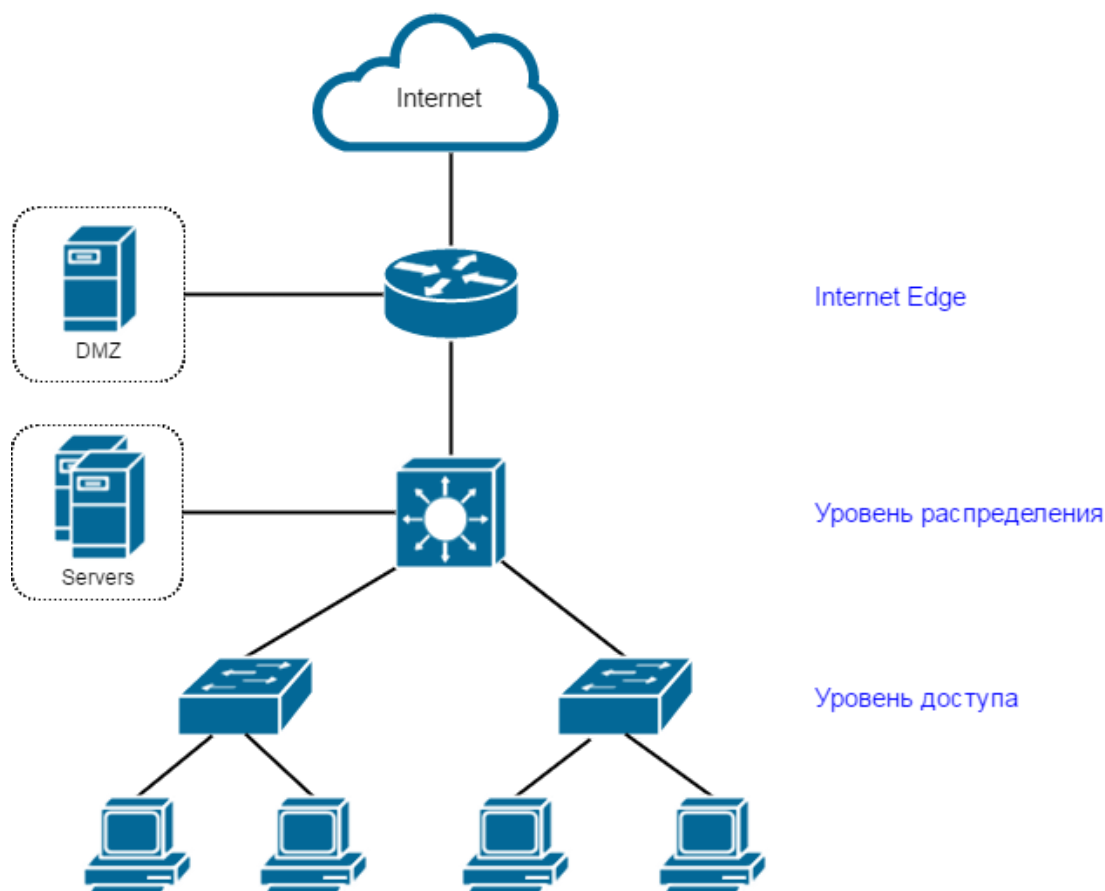
5.5 Чем защищать?

Определившись с тем, что защищать, от чего и как, остается последний вопрос - Чем? Ответ не так прост, как может показаться на первый взгляд.

Во-первых, мы можем организовать защиту как на сетевом уровне, так и на уровне операционной системы. И Windows и Linux имеют встроенные межсетевые экраны с помощью которых можно ограничить доступ аналогично access-list-ам на сетевом оборудовании. В то же время с помощью сетевого уровня мы можем задать правила доступа сразу на целый сегмент, без необходимости настройки на каждом отдельном компьютере или сервере. Что лучше выбрать? Мое мнение, что ограничение доступа на сетевом уровне - обязательно! Таким образом мы защищаем целые сегменты от таких угроз как сканирование портов или несанкционированный доступ. Настройка же межсетевых экранов на компьютерах (особенно на серверах) будет отличным дополнением, которое позволит защищаться даже внутри одного сегмента. Стоит понимать, что если взломанный компьютер пытается заразить соседний, то трафик не уходит дальше ближайшего L2 коммутатора. Соответственно списки доступа на маршрутизирующем оборудовании в данном случае бессильны.

Используйте правила доступа как на сетевом уровне, так и на уровне ОС.

Во-вторых, почти всегда возникает вопрос - на каком оборудовании применять списки доступа? Если вспомнить классическую схему офисной сети, то можно обнаружить маршрутизатор в качестве пограничного устройства и L3 коммутатор, как ядро сети.



Естественно, что данная схема лишь некий шаблон, к которому стоит стремиться. Таким образом защиту локальных серверов лучше организовывать на L3 коммутаторе, тем самым полностью изолировав их от пограничного устройства. Именно этот способ мы будем рассматривать далее. На пограничном же маршрутизаторе логичнее всего защищать публичные сервера, т.е. организовать DMZ. Об этом мы поговорим в 6-ой главе. Такой подход позволит избежать бардака в списках доступа. Настоятельно рекомендую к прочтению данную [статью](#), которая описывает разницу между маршрутизатором и L3 коммутатором.

5.6 Списки доступа

После большой преамбулы мы наконец добрались до ключевой темы этой главы - списки доступа или, как их еще называют, **access-list**-ы. Что они из себя представляют? Это правила, которые либо что-то разрешают, либо запрещают. Используются списки доступа как для классификации, так и для фильтрации трафика. Т.е. access-list-ы могут применяться не только для ограничения доступа (пакетная фильтрация), но и для NAT-а, QoS, Policy based routing, VPN и т.д. В рамках книги мы рассмотрим именно пакетную фильтрацию. Сами списки доступа бывают нескольких видов:

1. Стандартные
2. Расширенные
3. Динамические
4. Рефлективные

5. Временные

Если вы впервые сталкиваетесь с этими понятиями, то настоятельно рекомендую ознакомиться с 15-м видео [уроком](#) из [Курса молодого бойца](#). Здесь же мы не будем описывать каждый вид. Чаще всего используются либо **стандартные (standart)**, либо **расширенные (extended)** списки доступа. Стандартные списки доступа могут проверять только адреса источника трафика. Расширенные же в качестве параметров фильтрации используют следующие объекты:

- ip-адрес источника
- порт источника
- протокол
- ip-адрес получателя
- порт получателя

Естественно, что в большинстве случаев лучше использовать именно расширенные списки доступа, т.к. они позволяют настроить доступ более гранулярно. Но несмотря на такое количество параметров, очень часто можно встретить настройку следующего вида:

```
access-list 101 permit ip any any
```

Подобный список доступа разрешает абсолютно весь трафик и обычно используется когда администратору просто лень “заморачиваться”.

*Старайтесь не использовать правила доступа разрешающие весь трафик
(**access-list 101 permit ip any any**).*

Чтобы этого избежать необходимо проделать довольно серьезную работу, результатом которой станет “правильный” список доступа. В идеальном случае должен разрешаться только действительно необходимый трафик, а весь “лишний” - блокироваться. Добиться этого весьма трудно, но есть несколько способов, которые сделают эту работу чуть легче. Мы рассмотрим это далее, а также познакомимся с **best practices** по формированию access-list-ов.

5.6.1 Матрица прав доступа

На заре появления сетей вопросы информационной безопасности были далеко не на первом месте. Локальные сети были “маленькие”, сервисов и угроз для них еще меньше. Когда же появилась задача назначения прав доступа то “безопасники” преимущественно использовали так называемую “**матрицу прав доступа**”.

Матрица прав доступа является универсальной и наиболее гранулированной формой контроля доступа, которая прямо “в лоб” описывает для каждого пользователя набор конкретных операций, которые ему разрешается выполнять по отношению к каждому объекту. Пример классической матрицы доступа представлен ниже:

	R1	R2	R3	...
user1	m(1,1)	0	m(1,3)	...
user2	m(2,1)	0	m(2,3)	...

user3	0	m(3,2)	0	...
...

Здесь **R** - это ресурс/сервер/сервис до которого организуется доступ, **user** - пользователь, которым доступ предоставляется. Параметр **m** описывает сами права доступа.

Как было сказано выше, это наиболее гибкий и гранулярный способ предоставления доступа. Но что если в вашей организации хотя бы 100 пользователей, а количество сервисов более 20. Представьте табличку 100x20. Очевидно что такой подход весьма утопичен, а для крупных организаций (более 1000 пользователей) в принципе невозможен.

Уверен, что большинство догадывается как решить проблему огромной матрицы - уменьшить ее за счет создания групп пользователей и групп сервисов. В этом случае матрица "схлопывается" до разумных размеров. Предположим, что всех пользователей мы разбили всего на четыре группы: администрация, бухгалтерия, пользователи и гости. Если сервисов/ресурсов очень много, то их тоже можно "свернуть" в группы, но в небольших компаниях можно указывать их в матрице в явном виде. В итоге мы получаем нечто подобное:

	Интернет	Почта	Файлы	Портал	...
администрация	2	1	2	2	...
бухгалтерия	1	1	1	1	...
пользователи	1	1	0	1	...
гости	1	0	0	0	...

Здесь 0 - доступ отсутствует, 1 - доступ ограничен, 2 - полный доступ. Уровни доступа это обычные шаблоны access-list-ов, которые либо полностью запрещают, либо полностью разрешают, либо разрешают частично (ограниченное кол-во портов). Это только пример и на практике можно вводить гораздо большее кол-во уровней доступа. Что касается ресурсов/сервисов, то вы должны были определить их на этапе аудита (инвентаризации). Если внимательно посмотреть на табличку, то можно понять, что мы фактически сформировали упрощенные списки доступа до конкретного ресурса. У нас есть источник (группа пользователей), получатель (ресурс) и порты (уровень доступа).

Составьте матрицу прав доступа перед началом формирования access-list-ов.

Вы просто обязаны составить подобную матрицу доступа перед тем, как начать формировать access-list-ы. Глядя на нее вы сможете оперативно определиться с назначением прав. Кроме того, данная матрица должна быть обязательно утверждена директором компании, дабы у пользователей не возникали претензии именно к вам. При этом не забывайте актуализировать матрицу если приходится изменять права доступа до каких-либо объектов. В идеале все изменения начинаются именно с матрицы, а уже затем распространяются на сетевое оборудование.

5.6.2 Методы назначения прав. RBAC

Если смотреть на описанную выше матрицу, то список доступа можно разбить на три основные части: источник, получатель и сетевые порты. Как вы понимаете, в качестве получателя чаще всего используется сервер либо сервис компании. Здесь редко возникают вопросы, т.к. данная информация собирается в ходе инвентаризации. Чаще всего возникают трудности с сетевыми портами и источниками. Сетевые порты мы рассмотрим чуть позже, а данный параграф посвятим источникам.

Опять же, глядя на матрицу доступа мы видим, что в качестве источника почти всегда определяются пользователи. При этом мы уже обсудили, что для удобства гораздо лучше применять группы пользователей, а не каждого по отдельности. Но здесь появляется другая проблема - Как сформировать эти группы? Как определить к какой группе относится пользователь и какие права ему назначать? Для ответа на этот вопрос существуют специальные методы назначения прав. Вот несколько из них:

- 1) **Дискреционный метод доступа (DAC, Discretionary Access Control)** - избирательный или произвольный метод доступа;
- 2) **Мандатный метод доступа (MAC, Mandatory Access Control)** - принудительный метод доступа;
- 3) **Ролевой доступ (RBAC, Role-based Access Control)** - доступ на основе ролей.

На практике чаще всего применяются либо DAC, либо RBAC. Наиболее приближенным к реальной жизни является именно RBAC. Этот метод мы и рассмотрим более подробно.

Как видно из названия, основным свойством RBAC является использование “ролей”, а не просто групп. Понятие “роль” в данном случае ближе всего к “должности” или “кругу должностных обязанностей”. Т.к. на одной и той же должности могут работать несколько людей (например - бухгалтер), то одна “роль” может быть приписана сразу нескольким пользователям. Фактически мы формируем все те же группы, но на основании ролей.

Набор ролей в RBAC-системе обычно соответствует перечню различных должностей и отделов, существующих в компании. RBAC лучше всего работает в организациях, в которых существует четкое распределение должностных обязанностей. RBAC идеален для государственных структур, нефтегазовых компаний, финансовых учреждений (банки, страховые компании), учебных заведений, военных структур и других крупных организаций.

Разрешения приписываются ролям, а не отдельным пользователям или группам пользователей. А уже затем те или иные роли назначаются пользователю. Например в системе управления доступом в Банке, всем юристам присваивается роль “юрист”, менеджерам - “менеджер” и т.д. Процесс определения ролей должен включать тщательный анализ того, как функционирует организация, какой набор функций должен выполнять сотрудник и какими сервисами он может пользоваться. Уже после этого ролям назначаются права доступа, которые необходимы пользователям для выполнения своих служебных обязанностей.

Все пользователи, играющие одинаковую роль, имеют идентичные права. Если же сотрудник получает повышение или перевод на другую должность, то ему просто назначается новая роль. Если в компании вдруг меняется бизнес-процесс и

сотрудникам определенной должности требуются новые права, то достаточно изменить всего лишь одну роль, к которой принадлежат эти пользователи. Такой подход существенно упрощает администрирование прав доступа, без необходимости внесения персональных правок.

Управляйте доступом на основе ролей пользователей (RBAC).

5.6.3 VLAN и RBAC

На словах все звучит довольно просто, однако на практике обязательно возникают дополнительные вопросы. И самый частый вопрос - Как эти группы и их роли увязать с сетями? Ведь маршрутизатор не понимает роль “менеджер” или “бухгалтер”. В списках доступа в качестве источника трафика нам необходимо указывать ip-адрес или сеть.

В современных маршрутизаторах есть отличная функция - возможность указать группу в списках доступа. Так называемая **object group**, в которую вы можете включить несколько сетей или хостов. Однако если мы говорим об уровне распределения и списки доступа настраиваются на L3 коммутаторах, то там такая функция отсутствует. Именно здесь появляется VLAN.

VLAN - это отличный способ разбить всех пользователей на группы или роли. Вы можете определить бухгалтера в один VLAN, а менеджеров - в другой. При этом каждому VLAN-у соответствует своя сеть, которую мы и будем указывать в access-list-ax. Пример разделения пользователей:

№	Группа/Роль	VLAN	Сеть
1	Бухгалтерия	10	192.168.10.0/24
2	Менеджеры	11	192.168.11.0/24
3	Администрация	12	192.168.12.0/24
...

Если у пользователя меняется роль, то ему просто присваивается соответствующий VLAN. Думаю многие теперь по другому взглянут на полезность сегментации. Правильно сегментированная сеть это уже половина работы в создании защищенной ИТ инфраструктуры.

Сегментируйте сеть в соответствии с существующими ролями/группами.

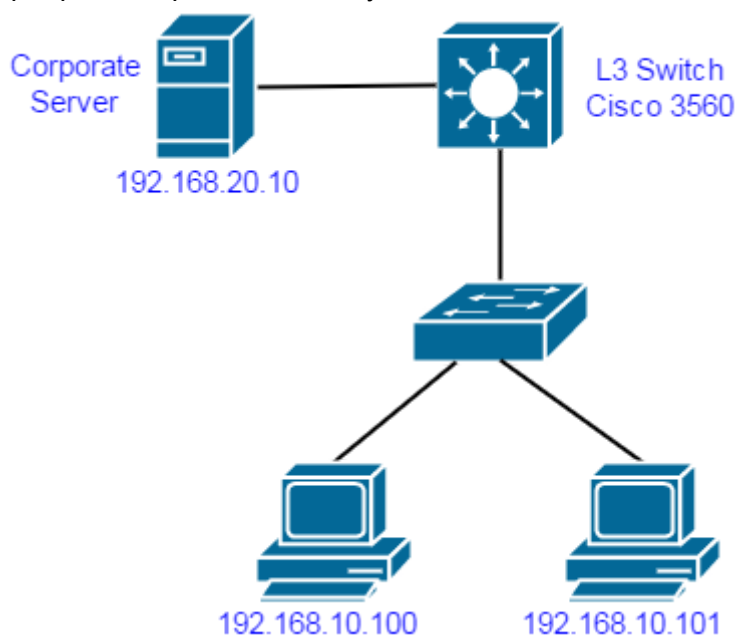
5.6.4 Порты

Мы добрались до третьего основного элемента в списках доступа - порты. Определив источник и получателя многие администраторы останавливаются на этом и разрешают абсолютно весь трафик. Это конечно лучше, чем вообще ничего, но все же не так безопасно. Основная проблема - как понять, какие порты открывать и для какого

протокола (tcp или udp)? Это не такая простая задача, как может показаться на первый взгляд, но все же выполнимая.

Начинать следует естественно с инвентаризации, которую вы должны были провести до этого. Имея перечень сервисов, которыми пользуются в вашей компании, вы сможете определить список минимально необходимых портов. Эти данные можно взять из описания самых сервисов (Email, Active Directory, DHCP, DNS, NTP и т.д.).

Однако, со стопроцентной долей вероятности у вас не удастся с первого раза создать идеальный список доступа. А если вы его примените, то рискуете получить частично неработающую сеть. Чтобы избежать этого, необходимо “подстраховаться” и проверить правильность своего списка. Сделать это можно с помощью логирования. Приведем пример и рассмотрим сеть следующего вида:



На уровне распределения (он же уровень ядра) установлен L3 коммутатор - Cisco 3560. VLAN 10 (192.168.10.0) определен для пользователей, а VLAN 20 (192.168.20.0) - для корпоративных серверов. Серверный сегмент организовать на L3 коммутаторе с целью обеспечения высокой пропускной способности. Нам необходимо узнать по каким портам пользователи обращаются к серверу 192.168.20.10. Для этого мы создадим разрешающий весь трафик access-list и включим для него логирование.

Настройки двух интерфейсов:

```
interface Vlan10
ip address 192.168.10.1 255.255.255.0
!
interface Vlan20
ip address 192.168.20.1 255.255.255.0
```

Создание списка доступа:

```
Switch(config)#ip access-list extended ForStatistic
Switch(config-ext-nacl)#permit ip any any log
```

Применение к интерфейсу:

```
Switch(config)#int vlan 10
Switch(config-if)#ip access-group ForStatistic in
```

Настройка лог-сервера:

```
logging facility local6  
logging 192.168.20.100          /ip-адрес лог-сервера
```

(Более подробно о настройке log-ов можно почитать во второй главе)

Таким образом когда пользователь будет обращаться к серверу, его трафик будет подпадать под заданный access-list. В результате будет генерироваться log-сообщение, с указанием ip-адреса и порта источника, ip-адреса и порта получателя и протокол. Пример:

```
Jun 10 10:30:15 router %SEC-6-IPACCESSLOGP: list ForStatistic permitted tcp 192.168.10.10(32791) -> 192.168.20.10(1080), 1 packet
```

Стоит отметить, что сообщение генерируется по первому совпадению, а затем по умолчанию одинаковые логи суммируются в одно событие в течении 5 минут. Интервал можно изменить при необходимости с помощью команды **ip access-list log-update threshold 10**. В результате, лог сообщение будет генерироваться после каждых 10 одинаковых событий.

В данном случае удобнее использовать именно внешний лог-сервер, т.к. объем логов может быть огромным. Фактически, нам необходим обычный syslog-сервер. Существует большое количество как бесплатных, так и платных вариантов.

Однако при большом количестве сообщений весьма трудно обработать их “вручную”. Для этого лучше использовать программы по анализу логов, которые позволяют в автоматическом режиме отображать наиболее часто используемые порты. Программы такого класса называются **Log Managment Systems**. Яркие примеры - Splunk, ELK, Graylog и т.д.

Используйте log-и для определения нужных портов.

В итоге вы сможете “увидеть” реальный трафик и заранее оценить адекватность планируемого списка доступа. Возможно вы пропустили какие-то сервисы, а возможно стоит ужесточить политику, тем самым урезав паразитный трафик, который не несет полезной нагрузки.

5.6.5 Входящий или исходящий

Еще одним важным моментом является применение списка доступа к интерфейсу. В предыдущем параграфе мы увидели пример команды (**ip access-group имя-списка in**). Думаю большинство читателей знают, что есть два варианта фильтрации трафика:

- 1) **Входящий (in)**. В этом случае трафик поступая на интерфейс L3 устройства сразу проверяется списком доступа и может быть отброшен, до того как начнется его обработка процессами маршрутизации. В некоторых случаях данный способ может существенно снизить нагрузку на процессор устройства.
- 2) **Исходящий (out)**. Трафик попадает в L3 устройство, обрабатывается, маршрутизируется и уже на выходе подвергается фильтрации. С точки зрения экономии ресурсов, это не самый выгодный способ.



Следует отметить, что в качестве интерфейса может выступать не только физический, но и виртуальный - VLAN (для роутеров это сабинтерфейсы).

В большинстве случаев рекомендуется фильтровать трафик именно на входе. Это и безопаснее и более эффективно в плане расхода ресурсов. По возможности старайтесь придерживаться этого метода. Однако в некоторых случаях гораздо удобнее фильтровать трафик именно на выходе, т.к. это существенно упрощает конфигурацию (ниже мы рассмотрим подобный пример). А если учитывать тот факт, что на уровне распределения используется L3 коммутатор, то обычно мы имеем хороший запас по производительности (чего не скажешь о маршрутизаторах).

*При выборе между **in** и **out** старайтесь придерживаться баланса между удобством и производительностью.*

Не забывайте, что к одному интерфейсу (будь то физический или виртуальный) вы можете присвоить два списка доступа - один на вход (in) и один на выход (out).

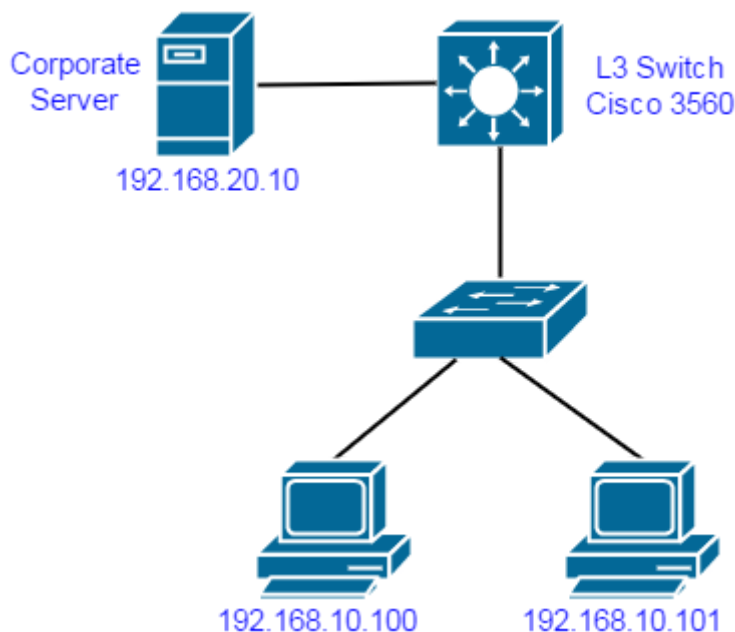
5.6.6 Два главных подхода

(Мы уже частично затронули эту тему в параграфе “Как защищать”, здесь же мы рассмотрим ее снова уже с точки зрения настроек)

Обычно список доступа состоит из нескольких правил. Как вы думаете, какое правило самое главное? **Последнее**. Как вам наверняка известно, правила работают по порядку - сверху вниз. Т.е. попав на интерфейс, пакет сначала проверяется первым правилом, затем (если оно не подошло) вторым и так до самого низа. И именно последнее правило определяет общий концепт. Как мы уже говорили, существует два основных подхода при создании списков доступа:

- 1) **Разрешено все, что не запрещено**. Т.е. в данном случае в качестве последнего правила указывается **permit ip any any**. Это значит, что вам нужно выше этого правила указать, что конкретно вы хотите запрещать.
- 2) **Запрещено все, что не разрешено**. Последнее правило - **deny ip any any**. По умолчанию весь трафик будет запрещен, а если что-то нужно разрешить, то это указывается выше.

Сразу может возникнуть логический вопрос: “Какой подход лучше?”. И здесь нет однозначного ответа. Каждый подход имеет право на существование и является более выгодным в той или иной ситуации. Естественно, что “Запрещено все, что не разрешено” - самый безопасный вариант. Однако в некоторых случаях его использование является затруднительным. Давайте приведем пример. Воспользуемся все той же схемой:



Предположим, что Corporate Server это FTP-сервер, к которому нужно разрешить только FTP и SSH протокол. Остальной же трафик до него должен быть запрещен. Т.е. если мы хотим придерживаться принципа “Запрещено все, что не разрешено”, то список доступа будет выглядеть примерно так:

```
ip access-list extended ForLocalSRV
 permit tcp 192.168.10.0 0.0.0.255 host 192.168.20.10 eq 21
 permit tcp 192.168.10.0 0.0.0.255 host 192.168.20.10 eq 22
 deny ip any any log
```

Однако если применить данный список на входящий трафик (**in**) на интерфейсе, который “смотрит” в сторону пользователей, то мы автоматически обязываем себя прописывать разрешающие правила и для выхода в Интернет. А это все же лучше делать на устройстве периметра (роутер, межсетевой экран). Казалось бы, что в данном случае придется придерживаться подхода “Разрешено все, что не запрещено”. Но в этом случае мы противоречим своим же рекомендациям! Есть способ обойти возникшую проблему. Все что нам нужно, это применить этот список доступа на исходящий трафик (**out**) уже на интерфейсе, который “смотрит” в сторону сервера.

```
Switch#conf t
Switch(config)#interface Vlan20
Switch(config-if)#ip access-group ForLocalSRV out
```

Это тот самый случай, когда фильтрация исходящего трафика гораздо удобнее и значительно упрощает конфигурацию, даже с использованием подхода “Запрещено все, что не разрешено”.

Возлагайте на устройства распределения только защиту локальных серверов.

5.7 Лучшие практики

Мы уже обсудили основные вопросы по формированию списков доступа, а именно: как определить источник, получателя, используемые порты, направление

фильтрации трафика и выбор общего концепта защиты. Однако все это не гарантирует оптимального решения. Как было сказано ранее, безопасность это не результат, а непрерывный процесс. Тоже самое можно сказать и о списках доступа. Если вы хотите получать от них максимум, то вы просто обязаны заниматься периодической проверкой.

Периодически проверяйте и актуализируйте списки доступа.

Кроме того, в следующих нескольких параграфах мы обсудим лучшие практики по составлению и применению access-list-ов. Придерживаясь их вы сможете существенно повысить эффективность ваших списков доступа.

5.7.1 Именованные списки доступа

Если вы начнете искать примеры настройки access-list-ов в сети Интернет, то с большой долей вероятности вы наткнетесь на инструкции вроде этой:

```
Switch(conf)# access-list 1 deny 10.0.3.2
Switch(conf)# access-list 1 permit any
```

Как видно, в качестве имени используется цифра (**1** в нашем примере). При этом если использовать цифру от 1 до 99, то это стандартный список доступа, а если от 100 до 199 - расширенный. Чем расширенный список доступа отличается от стандартного мы рассмотрели в пункте 5.6. Многие до сих пор используют подобный метод.

Однако, практически во всех прошивках (начиная с версии 12) для L3-коммутаторов появилась возможность использовать **именованные списки доступа**. Название говорит само за себя. Вместо абстрактной цифры мы можем использовать конкретное имя, которое будет однозначно характеризовать предназначение этого списка доступа. Пример:

```
Switch(conf)# ip access-list extended ForCorpSrv
Switch(config-ext-nacl)# permit tcp host 10.0.3.2 host 192.168.4.10 eq 3389
Switch(config-ext-nacl)# permit tcp host 10.0.3.2 host 192.168.4.10 eq 80
Switch(config-ext-nacl)# deny ip any any log
```

Согласитесь, что по имени **ForCorpSrv** гораздо проще понять, для чего используется этот список доступа, чем если бы мы использовали цифру 100.

Используйте именованные списки доступа.

Применение именованных списков доступа значительно упростит работу администратора, т.к. он сможет быстрее ориентироваться в конфигурации устройства и вносить необходимые правки.

5.7.2 Логирование правил

В пункте 5.6.4 мы упомянули возможность логирования для списков доступа. Смысл заключается в простом добавлении команды **log** в конец правила. В результате, когда трафик попадает под соответствующее правило (в английской литературе это называется **hit**) генерируется syslog-сообщение. Данный подход можно

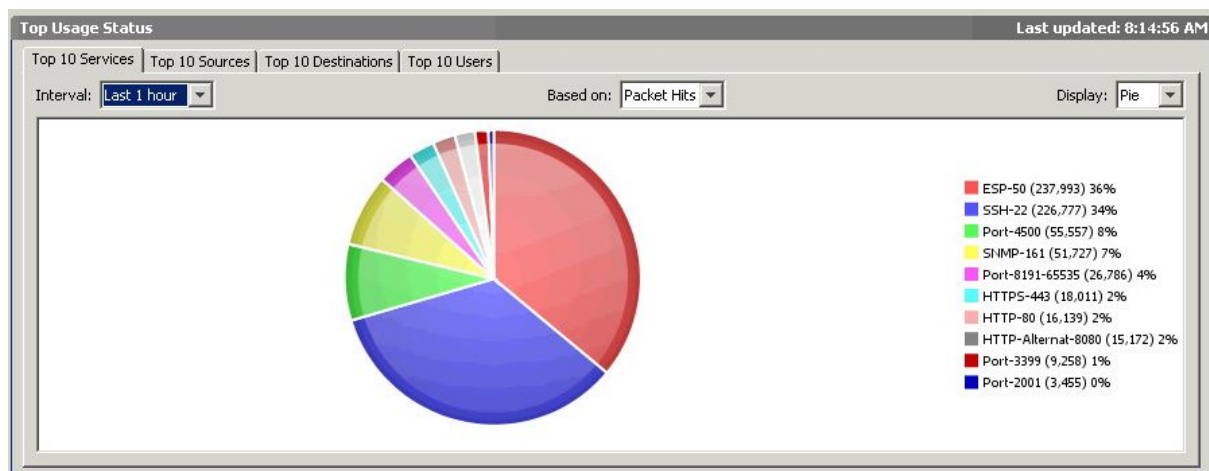
использовать не только в процессе составления списков доступа, но и в последующем анализе их эффективности. Автор настоятельно рекомендует включать логирование для правил на постоянной основе, или хотя бы периодически. Естественно, что это обяжет вас использовать внешний Лог-сервер, а еще лучше системы анализа логов, о которых мы говорили ранее. Приемы оптимизации правил мы рассмотрим в следующих параграфах.

Используйте логирование правил для их оптимизации.

Если вы придерживаетесь подхода “Запрещено все, что не разрешено”, то вы просто обязаны включить логирование хотя бы для последнего запрещающего правила - **deny ip any any log**. Это позволит вам оперативно отслеживать не блокируется ли нужный трафик в случае каких-либо проблем с сетью.

5.7.3 Топ-правила

Включив логирование для всех правил, мы можем получать очень важную статистику по их использованию. Как мы уже говорили ранее, правила работают по порядку - сверху вниз. Это значит, что если у вас 20 правил в списке доступа, то пакет будет проверяться всеми по очереди, пока не будет совпадения. Если пакет не найдет совпадений, то он упрется в последнее, запрещающее правило (**deny ip any any log**). В случае большого трафика, подобный перебор правил может серьезно загружать оборудование. Для снижения нагрузки вы можете составлять “короткие” списки доступа с небольшим количеством правил, однако это не всегда возможно. Гораздо логичнее будет размещение часто используемых правил наверху списка. Это может значительно снизить нагрузку, т.к. большая часть пакетов будет попадать под первое либо второе правило. Обнаружить такие Топ-правила как раз и позволяет включенное логирование. Чем больше хитов (срабатываний) у правила, тем выше оно должно быть в общем списке. Пример статистики:



Если же вы все же не хотите включать логирование (или нет такой возможности), вы можете периодически проверять статистику по правилам используя следующую команду:

```
Switch#show access-lists
Extended IP access list TEST
10 permit icmp host 192.168.20.10 host 192.168.10.100 (4 match(es))
```

```
20 permit tcp host 192.168.20.10 host 192.168.10.100 eq www (5 match(es))
30 deny ip any any
```

Это всего лишь пример, который показывает возможность оценить статистику по использованию правил. Чем больше **matches**, тем выше правило.

Помещайте часто используемые правила в самый верх списка доступа.

К примеру логично предположить, что составляя access-list для Web-сервера самым первым правилом лучше разрешить доступ по 80 порту, а уж затем все остальное.

5.7.4 Неиспользуемые правила

По работе мне весьма часто приходилось встречать очень длинные списки доступа (30 и более правил). Начиная проверять, оказывалось, что половина из них уже давно не нужна, т.к. одного из серверов уже нет или часть правил просто дублируют друг друга. Новые правила дописывались годами, при этом никто не спешил удалять старые. Как мы уже выяснили ранее, чем больше правил, тем больше ресурсов необходимо на обработку пакета, даже если он и вовсе должен быть запрещен. Чтобы избежать бардака и расходования ресурсов устройства, необходимо удалять неиспользуемые правила. В этом вам опять же могут помочь анализаторы логов, либо все та же команда **show access-lists**. Логично, что если в течении недели по правилу нет ни одного хита (совпадения), то его с большой долей вероятности можно удалить.

Удаляйте неиспользуемые правила.

5.7.8 Паразитный трафик

Практически в любой сети огромное количество “паразитного” трафика. Это такие сервисы как domain-udp, bootp, NetBios и т.д. Как правило такие пакеты попадают на L3 устройство в виде широковещательных запросов. Их количество сильно зависит от вашей сети и используемых сервисов. Все эти пакеты также будут обрабатываться вашими правилами доступа. К тому же данный трафик будет генерировать огромное количество бесполезных логов. Чтобы это избежать, по возможности постарайтесь определить какой трафик для вашей сети является паразитным, создайте для него специальные правила и не включайте логирование. Данные правила лучше поместить как можно выше, дабы отбрасывать подобные пакеты сразу и не “протаскивать” их через все остальные.

Отключите логирование для паразитного трафика.

Не могу не привести пример из реальной жизни от моего коллеги Дениса Тимошенко. О его опыте создания политики межсетевого экранирования он подробно рассказал в своем [блоге](#). Если статья вдруг будет не доступна, вы можете скачать ее в [pdf](#) формате. Настоятельно рекомендую к прочтению. Вы будете удивлены, узнав, как

можно разгрузить каналы передачи данных, только лишь грамотно сформированными списками доступа.

5.8 Обратные списки доступа

Как вы наверняка знаете, почти все сетевое взаимодействие является двухсторонним. Это значит, что сделав запрос, мы обязательно получим ответ. Так работает **tcp** и **icmp**. Однако, до сих пор мы рассматривали списки доступа только для трафика в одну сторону - до получателя. Но как быть с обратным трафиком? По хорошему для него тоже необходимо создать список доступа. Однако на практике это весьма трудно сделать. Дело в том, что осуществляя запрос на конкретный адрес и конкретный порт получателя, порт источника может быть выбран произвольным образом (либо из определенного диапазона портов). В большинстве случаев это делает невозможным формирование обратных списков доступа. Именно по этой причине на L3 коммутаторе обычно организывают защиту только для локальных серверов. В локальной сети основным источником угрозы является именно компьютер пользователя, а для защиты от них нам достаточно списков доступа в одну сторону - до серверов.

Для публичных же серверов и сегмента Интернет этого недостаточно и требуется двусторонняя фильтрация. Осуществить это можно лишь с помощью технологии **stateful inspection**, о которой мы и поговорим в следующей, заключительной главе.

5.9 Чеклист №5

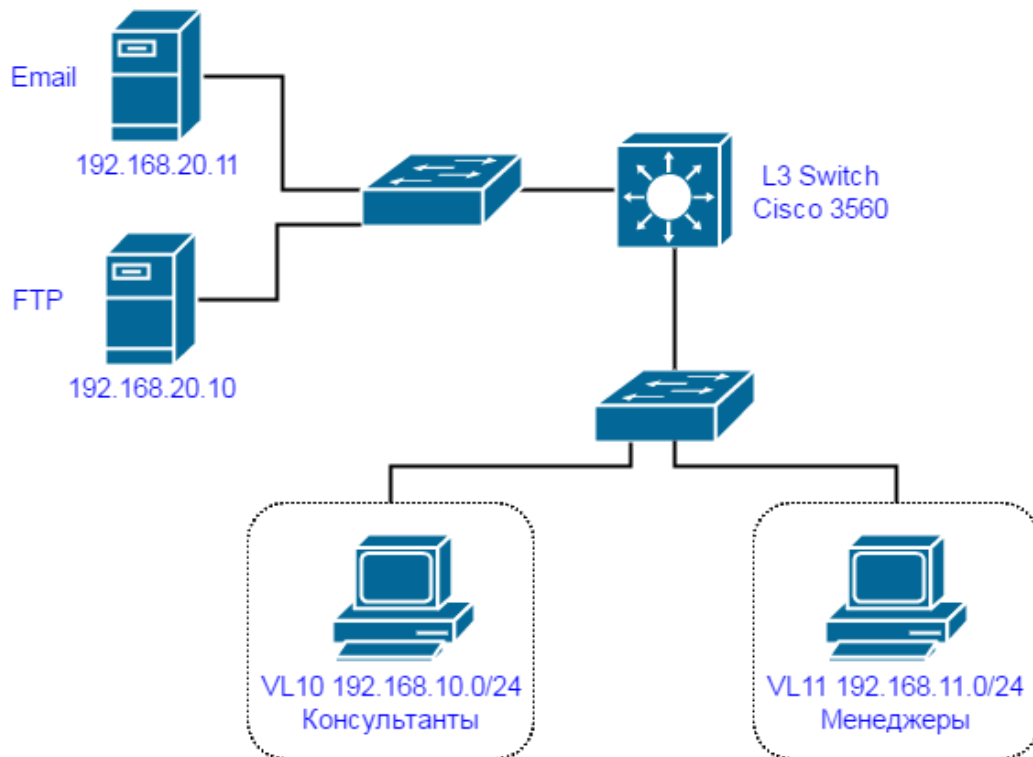
Пятая глава подошла к концу. Хотелось бы напомнить, что мы обсуждали вопрос формирования списков доступа на уровне распределения. О защите периметра мы поговорим в следующей главе. Чеклист №5:

1. *Не оставляйте сегменты без соответствующих списков доступа.*
2. *Проведите “аудит” ИТ-инфраструктуры перед формированием списков доступа.*
3. *Составьте перечень наиболее важных/критичных систем и сервисов.*
4. *Определите от чего и от кого вы будете защищаться.*
5. *При создании списков доступа старайтесь придерживаться метода “Запрещено все, что не разрешено”.*
6. *Используйте правила доступа как на сетевом уровне, так и на уровне ОС.*
7. *Старайтесь не использовать правила доступа разрешающие весь трафик (**access-list 101 permit ip any any**).*
8. *Составьте матрицу прав доступа перед началом формирования **access-list**-ов.*
9. *Управляйте доступом на основе ролей пользователей (**RBAC**).*
10. *Сегментируйте сеть в соответствии с существующими ролями/группами.*
11. *Используйте **log**-и для определения нужных портов.*
12. *При выборе между **in** и **out** старайтесь придерживаться баланса между удобством и производительностью.*

13. Возлагайте на устройства распределения только защиту локальных серверов.
14. Периодически проверяйте и актуализируйте списки доступа.
15. Используйте именованные списки доступа.
16. Используйте логирование правил для их оптимизации.
17. Помещайте часто используемые правила в самый верх списка доступа.
18. Удаляйте неиспользуемые правила.
19. Отключите логирование для паразитного трафика.

5.10 Пример конфигурации

В качестве пример давайте рассмотрим сильно упрощенную, классическую схему с L3 коммутатором на уровне распределения (сегмент выхода в Интернет здесь не рассматривается):



В нашем случае не требуется инвентаризация, т.к. из серверов у нас всего два: Email и FTP. Кроме того я уже разбил всех пользователей на две группы: Консультанты и Менеджеры. Матрица доступа будет выглядеть примерно так:

	Email-сервер	FTP-сервер
Консультанты	1	0
Менеджеры	1	1

где 0 - нет доступа, 1 - есть, но ограниченный.

Сразу условимся, что не будет настраивать на L3-коммутаторе правила для выхода в Интернет. В связи с этим нам гораздо выгоднее применять список доступа в направлении **out** на порту, который "смотрит" в сторону серверов. В нашем случае это

Vlan20 (сеть 192.168.20.0/24). Также мы будем придерживаться подхода “Запрещено все, что не разрешено”.

Также не потребуется предварительный анализ логов на предмет используемых портов, т.к. в нашем случае это всего лишь три порта: **FTP (tcp/21)**, **SMTP (tcp/25)** и **POP3 (tcp/110)**. Для наглядности давайте еще разрешим **icmp** трафик. Кроме того, мы хотели бы отбрасывать все **NetBios (tcp/139)** пакеты и не логировать их, дабы не портить “картину”.

Итак, мы имеем матрицу доступа, знаем источники, получателей, порты по которым будет взаимодействие, определились с направлением фильтрации и общим концептом защиты. Можно приступать к формированию списков доступа.

```
Switch(config)#ip access-list extended ForLocalSRV
Switch(config-ext-nacl)#deny tcp 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 139
Switch(config-ext-nacl)#deny tcp 192.168.11.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 139
Switch(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 21 log
Switch(config-ext-nacl)#permit tcp 192.168.11.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 21 log
Switch(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 25 log
Switch(config-ext-nacl)#permit tcp 192.168.11.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 25 log
Switch(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 110 log
Switch(config-ext-nacl)#permit tcp 192.168.11.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 110 log
Switch(config-ext-nacl)#deny ip any any log
```

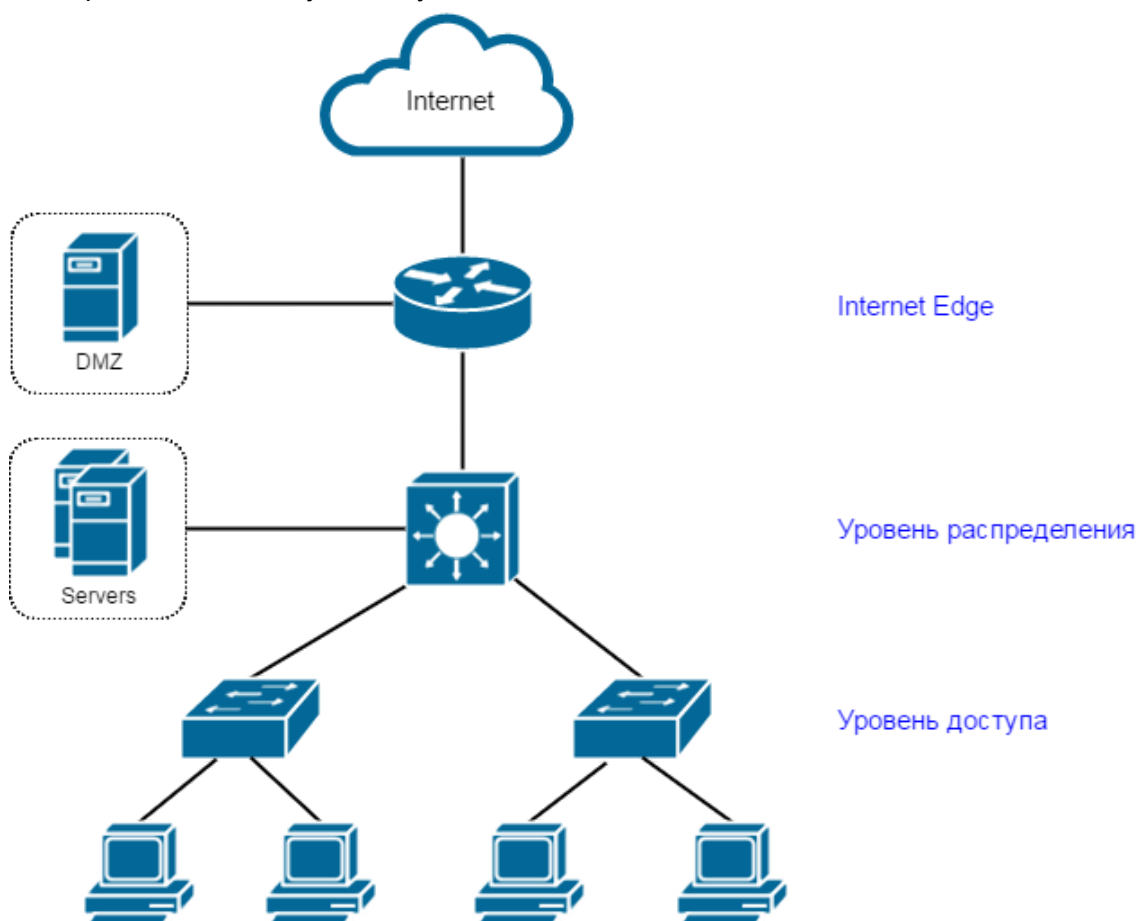
Затем применяем к интерфейсу:

```
Switch(config)#int vlan 20
Switch(config-if)#ip access-group ForLocalSRV out
```

Для настройки внешнего Лог-сервера можете воспользоваться второй главой книги. Как видите, это очень упрощенный пример, но в целом он отражает большую часть процесса.

6. Защита периметра

Вот мы и добрались до заключительной главы в этой книге. Мы обсудили большинство аспектов по защите локальной сети от угроз, которые могут возникнуть изнутри, будь то нерадивый пользователь, либо хакер, который уже проник внутрь. Теперь же мы обсудим каким образом защититься от угроз из сети Интернет. Т.е. если рассмотреть классическую схему ниже:



мы сосредоточимся на модуле периметра (**Internet Edge**) и защите **DMZ**. Большинство уже описанных рекомендаций по защите сети актуальны и в данном случае. Однако безопасность периметра это совершенно отдельная задача, которая требует своих подходов и технологий.

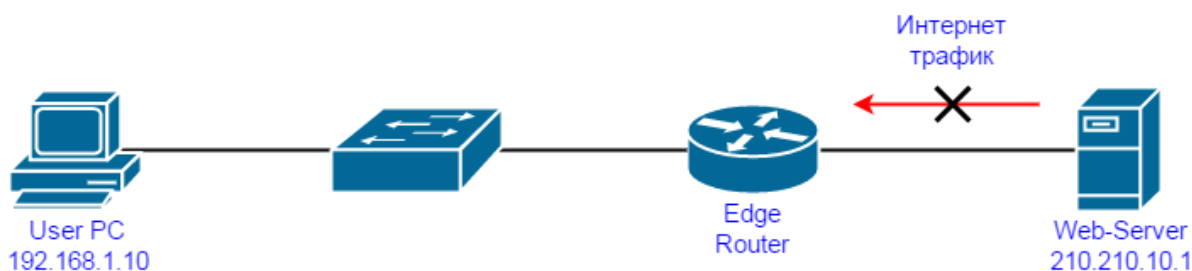
В этой главе мы как обычно рассмотрим лучшие практики и приведем несколько показательных примеров. Но самую главную идею этой главы можно выразить всего лишь одним предложением:

Минимизируйте разрешенный трафик из сети Интернет.

Далее мы подробно рассмотрим, каким образом можно этого добиться.

6.1 Stateful Packet Inspection

В параграфе 5.8 мы затронули сложности создания обратных списков доступа. Давайте рассмотрим эту тему немного подробнее. Предположим на периметре сети стоит обычный маршрутизатор:



Если придерживаться главного принципа по защите периметра, то весь трафик из сети Интернет должен быть запрещен. Т.е. в простейшем случае настройки должны выглядеть как в примере ниже.

Настройка интерфейсов:

```
interface GigabitEthernet0/0
description outside
ip address 210.210.10.2 255.255.255.0
ip access-group Outside-Inside in
ip nat outside
!
interface GigabitEthernet0/1
description inside
ip address 192.168.1.1 255.255.255.0
ip nat inside
```

Настройка NAT:

```
ip access-list extended ForNAT
permit ip 192.168.1.0 0.0.0.255 any
```

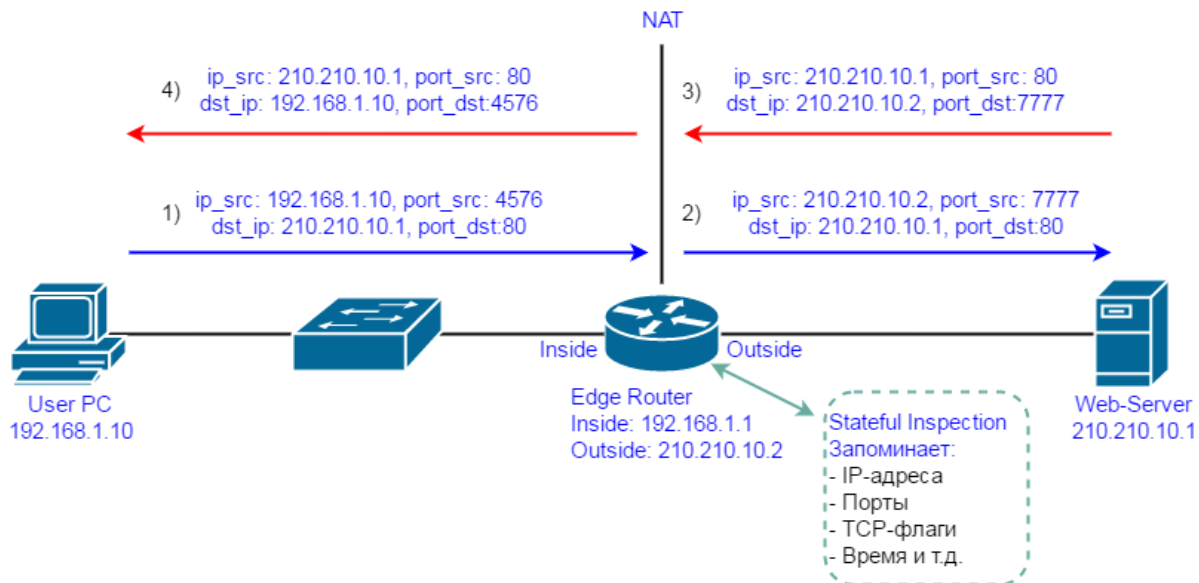
Список доступа:

```
ip access-list extended Outside-Inside
deny ip any any log
```

Список доступа **deny ip any any** будет блокировать любой трафик входящий из сети Интернет на внешний интерфейс маршрутизатора. Казалось бы, задача выполнена. Но, как многие уже наверно догадались, Интернет так работать не будет (если не верите, то можете собрать макет в Cisco Packet Tracer). Как мы уже говорили ранее, почти все сетевое взаимодействие является двусторонним. Сделав запрос на Web-сервер (по протоколу HTTP/HTTPS) мы обязательно должны получить ответ. Также работают и другие сервисы - запрос/ответ. Т.е. хотим мы или нет, но обратный трафик из Интернета необходимо пропускать. Но здесь мы опять сталкиваемся с проблемой описанной в пункте 5.8 - динамический выбор порта источника и, как следствие, невозможность сформировать конкретный access-list, который будет пропускать обратный трафик. Разрешать же обратный трафик по всем возможным портам тоже не вариант.

Решить нашу проблему обратных списков доступа можно с помощью технологии **SPI (Stateful Packet Inspection)**. Если перевести дословно, то это

инспекция пакетов с хранением состояния. Данная тема весьма важна, поэтому мы подробно рассмотрим работу этой технологии. Для этого еще раз рассмотрим описанный ранее пример:



1) На первом этапе компьютер пользователя формирует http запрос на Web-сервер. Содержимое пакета будет примерно следующим:

IP адрес источника - 192.168.1.10, порт - 4576

IP адрес получателя - 210.210.10.1, порт - 80

Порт получателя вполне предсказуем (http/80), а вот порт источника, как видите, был выбран динамически (4576 в нашем случае).

2) Согласно маршруту по умолчанию пакет попадает на роутер. Помимо стандартных процессов маршрутизации начинает работу технология SPI. Роутер запоминает данные о пакете (адреса/порты источника и получателя, tcp флаги, время запроса и т.д.), а затем с помощью NAT-а передает пакет в сеть Интернет. Здесь стандартная процедура, NAT заменяет "серый" адрес источника на "белый", а также динамически изменяет порт источника (в нашем случае на 7777 порт). Параметры получателя остаются неизменными. На выходе с роутера получаем следующий пакет:

IP адрес источника - 210.210.10.2, порт - 7777

IP адрес получателя - 210.210.10.1, порт - 80

3) Сервер получает запрос и формирует ответ. Ответ в данном случае будет абсолютно зеркальным:

IP адрес источника - 210.210.10.1, порт - 80

IP адрес получателя - 210.210.10.2, порт - 7777

4) Ответный пакет приходит на маршрутизатор. Снова отрабатывает NAT, который определяет, что порту 7777 соответствует ранее созданный запрос от локального компьютера с адресом 192.168.1.10. Пакет транслируется следующим образом:

IP адрес источника - 210.210.10.1, порт - 80

IP адрес получателя - 192.168.1.10, порт - 4576

Увидев, что пакет предназначен для локальной сети, маршрутизатор проверяет свои SPI записи. Обнаруживая, что данный пакет является ответом на ранее созданный запрос, пакет пропускается к получателю.

Другими словами, Stateful Packet Inspection позволяет определить, что сессия была инициирована из локальной сети, а следовательно обратный трафик нужно пропустить, даже без наличия специализированных access-list-ов. При этом любой другой трафик, который инициируется из внешней сети, будет блокироваться.

*Обязательно используйте технологии **Stateful Inspection** на периметре сети.*

Изначально, технологию Stateful Packet Inspection придумала компания **Check Point**. Затем ее “подхватили” и другие. Долгое время возможности SPI присутствовали только на межсетевых экранах. Именно поэтому на границе сети всегда рекомендуют ставить межсетевой экран вместо классического маршрутизатора (в чем главные различия между МЭ и маршрутизатором можно почитать [здесь](#)). **Cisco ASA** является ярким представителем межсетевых экранов, где в полной мере реализуется технология Stateful Inspection. Однако, еще в самом начале книги мы обусловились, что не обладаем специализированными средствами защиты. Что же делать, если в нашем распоряжении обычный роутер Cisco? Использовать технологию **Zone-Based Firewall**.

6.2 Zone-Based Firewall

Zone-Based Firewall (ZBF) - способ создания межсетевого экрана на основе маршрутизатора Cisco (фактически, это реализация технологии Stateful Inspection). Функционал ZBF впервые появился в Cisco IOS версии **12.4(6)T** и на сегодняшний момент поддерживается большинством маршрутизаторов Cisco:

- Cisco ISR (Cisco 800 Series Routers; Cisco 1800, 2800 и 3800)
- Cisco 7200 Series Routers
- Cisco 7301 Router
- Cisco ASR 1000 Series

На самом деле это не первая попытка компании Cisco сделать межсетевой экран из маршрутизатора. Еще раньше появилась функция **CBAC (context-based access control)**, которая также использует технологию stateful inspection. Вы можете посмотреть его настройку в [17](#) уроке Курса молодого бойца. Однако ZBF более новая технология, которая продолжает развиваться (в отличии от CBAC).

*Убедитесь, что прошивка маршрутизатора поддерживает **Zone-Based Firewall**.*

Ключевое отличие Zone-Based Firewall - управление доступом на основе зон безопасности, вместо классической модели на основе интерфейсов. Рассмотрим более детальное сравнение с CBAC:

CBAC	ZBF
Настройка на основе интерфейса	Настройка на основе зон
Управление входящим и исходящим доступом к интерфейсу	Управление доступом между зонами в двух направлениях
Настройка роутеров с более чем двумя интерфейсами может стать достаточно сложной	Простая настройка роутеров с более чем двумя зонами

Используются списки доступа с сохранением состояния	Используется основанный на политике классов Cisco Common Classification Policy Language
Проверка и контроль на уровне приложения не поддерживается	Поддерживается проверка и контроль на уровне приложения
Поддерживается начиная с IOS Release 11.2	Поддерживается начиная с IOS Release 12.4 (6) T

Несмотря на кажущуюся сложность, настройка Zone-Based Firewall сводится к простым шести шагам:

- 1) Определение зон (**zone**);
- 2) Определение пар зон (**zone pair**);
- 3) Определение интересующего трафика (**class map**);
- 4) Определение политики (действий) в отношении выбранного трафика (**policy map**);
- 5) Применение политики к парам зон (**service policy**);
- 6) Определение интерфейсов маршрутизатора в зоны.

Для наглядности, в следующих параграфах мы рассмотрим каждый пункт, а в конце главы как обычно будет приведен пример конфигурации. Следует заметить, что ZBF частично доступен в Cisco Packet Tracer, однако в GNS3 или EVE NG вы получите более полный функционал.

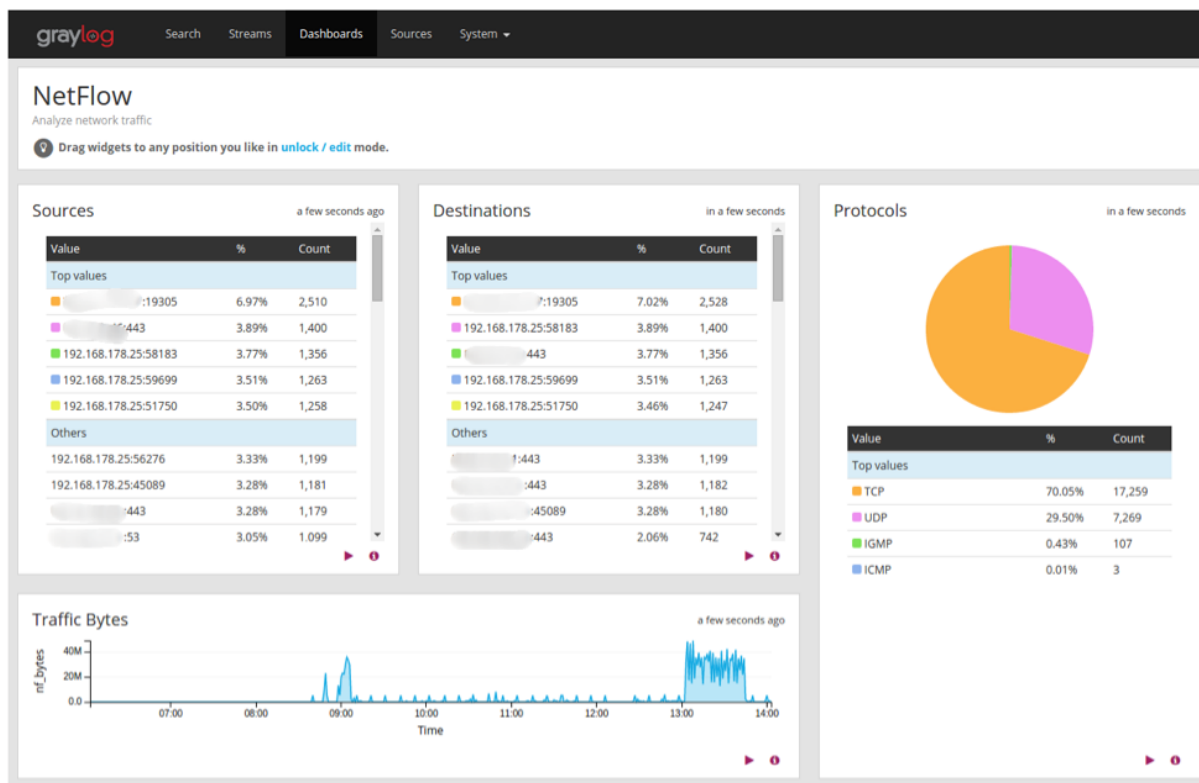
6.2.1 Подготовка (NetFlow)

Перед началом описания процесса настройки Zone-Based Firewall хотелось бы обсудить еще одну не менее важную вещь - подготовка. Как вы помните из главы 5, перед началом создания списков доступ необходимо провести предварительный аудит и сформировать матрицу доступа. Данные рекомендации актуальны и для защиты периметра. Перед началом настройки ZBF вы должны понимать, что вы собираетесь делать. Это самая частая ошибка администраторов, когда начинается настройка интернет шлюза (будь то маршрутизатор или межсетевой экран) без актуальной матрицы доступа. Не делайте так. Начните с подготовки и уже затем переходите к конфигурации устройства.

Для определения сервисов и ресурсов, которые посещают ваши пользователи, вы также можете воспользоваться рекомендациями по логированию (пункт 5.6.4 Порты). Кроме того, большинство маршрутизаторов поддерживают протокол **NetFlow**, который сможет предоставить вам подробнейшую статистику по вашему интернет трафику. Говоря про NetFlow нужно понимать три сущности:

- 1) **Сенсор**. В нашем случае в качестве сенсора будет использоваться сам маршрутизатор. Именно с него мы будем снимать статистику (“дамп”) трафика.
- 2) **Коллектор**. NetFlow данные нужно собирать в одно место (особенно если устройств несколько). Для этого и применяется NetFlow коллектор. В качестве коллектора может выступать обычный Linux сервер с установленным бесплатным **nfdump**.
- 3) **Анализатор**. Сами NetFlow данные в “сыром” виде плохо читаемы, как и в случае с логами (особенно при больших объемах). Чтобы получить ценную информацию, нужен анализатор этих данных. Для этого используются NetFlow анализаторы. Их также огромное количество. Есть бесплатные (**nfsen**, **graylog**),

есть платные, но с бесплатным триальным режимом (**Splunk, SolarWinds, ManageEngine**). Большинство анализаторов могут выступать и в качестве коллекторов. После анализа мы можем получить различные отчеты в графическом виде, как на картинке ниже:



*Проведите аудит с помощью **NetFlow** и составьте матрицу доступа для периметра.*

На основе этой информации вы сможете более детально проработать матрицу доступа, разрешив только действительно нужный трафик. Не пренебрегайте этим процессом!

Мы не будем рассматривать настройки NetFlow коллекторов и анализаторов, а вот настройки маршрутизатора опишем. В первую очередь конфигурируется интерфейс, с которого мы хотим снимать статистику по трафику:

```
Router(config)# interface FastEthernet 0/1
Router(config-if)# ip flow ingress
Router(config-if)# ip flow egress
```

Затем указывается ip адрес NetFlow коллектора (он же может быть и анализатором), UDP порт (на котором “слушает” коллектор) и версия NetFlow протокола:

```
Router(config)# ip flow-export destination 192.168.0.100 2055
Router(config)# ip flow-export version 5
```

Для отладки можно воспользоваться командой **show ip cache flow**, которая выдает локальную статистику netflow прямо на маршрутизаторе. С помощью **show ip flow export** можно увидеть статистику по взаимодействию с коллектором (сколько потоков

отправлено, сколько пакетов ушло на коллектор, ошибки при взаимодействии с коллектором и т.п.).

6.2.2 Zone

Мы уже дважды затрагивали тему сегментации и оба раза в локальной сети. Теперь очередь за периметром. Сегментация на пограничном устройстве не менее важна. Однако в данном случае речь идет не о физических интерфейсах и не о VLAN-ах. Под сегментами мы будем понимать зоны, для которых отличаются уровни безопасности. Количество зон зависит исключительно от размеров и сложности ИТ-инфраструктуры компании, однако почти всегда можно выделить три основные зоны:



- 1) **Outside** - внешняя сеть. Обычно это Интернет или же какой-либо другой недоверенный канал.
- 2) **Inside** - все что касается внутренних сетей (т.е. “локалка” компании).
- 3) **DMZ** - сегмент публичных серверов, к которым необходим доступ из Outside сети (т.е. обычно из Интернета).

Естественно, что названия всех зон могут быть другими, главное, чтобы вы могли понять их предназначение по имени.

Определите зоны безопасности на устройстве периметра.

В отличие от VLAN-ов, весь трафик между разными зонами по умолчанию **запрещен**. Это одно из ключевых отличий межсетевого экрана от обычного роутера. Таким образом мы автоматически получаем подход “Запрещено все, что не разрешено”, а это значит, что мы обязаны создавать правила доступа. Как это делается мы обсудим в следующих параграфах.

Сами зоны создаются элементарно:

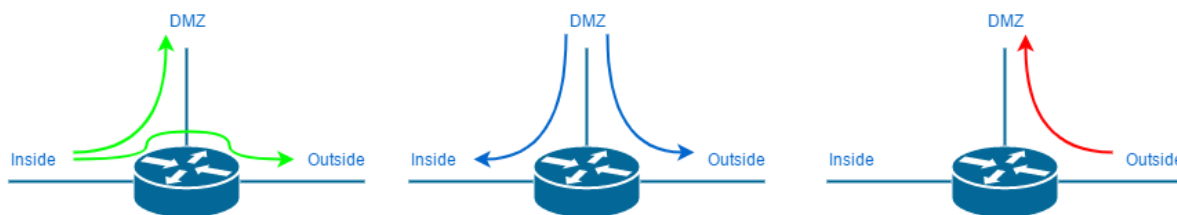
```
Router(config)#zone security Outside
Router(config-sec-zone)#exit
Router(config)#zone security Inside
Router(config-sec-zone)#exit
Router(config)#zone security DMZ
Router(config-sec-zone)#exit
```

Следует отметить, что в одну зону могут входить сразу несколько интерфейсов (или сабинтерфейсов) маршрутизатора. При этом трафик в рамках одной зоны по умолчанию разрешен.

6.2.3 Zone-pair

Как вы помните, access-list-ы применяются к интерфейсам. В ZBF все немного иначе. Здесь списки доступа применяются к так называемым “парам”, т.е. к трафику

между двумя зонами. Таким образом, перед началом создания правил вы должны определить все возможные пары зон, между которыми должен “ходить” трафик. Опять же, количество таких пар сильно зависит от вашей инфраструктуры, но почти всегда можно выделить пять основных, как на рисунке ниже:



- 1) Пара **Inside_Outside**. Трафик из внутренней сети во внешнюю. Обычно под внешней сетью подразумевается Интернет.
- 2) Пара **Inside_DMZ**. Трафик из внутренней сети в DMZ, т.е. к публичным серверам компании. В DMZ чаще всего располагают веб-сервера, почтовые сервера, DNS и т.д. Естественно что доступ к ним должен быть не только из Интернета, но и из внутренней сети.
- 3) Пара **DMZ_Outside**. Трафик от серверов DMZ во внешнюю сеть. Чаще всего используется для каких-либо обновлений.
- 4) Пара **DMZ_Inside**. Трафик от серверов DMZ во внутреннюю сеть. В данном случае обычно идет речь о взаимодействии с такими внутренними ресурсами как NTP, Active Directory и т.д. К защите данной пары стоит подходить с особой серьезностью, т.к. если злоумышленнику все же удастся взломать публичные сервера, следующей целью для него будет именно внутренняя сеть.
- 5) Пара **Outside_DMZ**. Именно здесь настраивается доступ к публичным серверам из внешней сети. Данной паре также необходимо уделить максимальное внимание, дабы не разрешить “лишнего”.

Настройка пар зон (zone pair) также довольно проста (пример):

```
Router(config)#zone-pair security Inside_Outside source Inside destination Outside
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Inside_DMZ source Inside destination DMZ
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security DMZ_Outside source DMZ destination Outside
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security DMZ_Inside source DMZ destination Inside
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Outside_DMZ source Outside destination DMZ
```

Определите все возможные пары зон, между которыми должен “ходить” трафик.

Хорошо подумайте перед выбором пар, возможно некоторые вам просто не нужны и вы вообще можете ограничиться одной - **Inside_Outside**. Чем меньше пар - тем безопаснее.

6.2.4 Class-map

С помощью **zone-pair** мы определили к каким направлениям трафика мы будем применять политики доступа. Следующим шагом будет выделение интересующего трафика. Делается это с помощью **class-map**. При этом для выделения трафика мы можем использовать несколько фильтров:

- 1) **Protocol** - это протоколы уровня 4 (TCP, UDP, ICMP), а также прикладные сервисы, такие как HTTP, SMTP, DNS, и т.д. Может быть указан любой известный или определяемый пользователем сервис.
- 2) **Access-group** - стандартный, расширенный или именованный ACL который может фильтровать трафик на основании IP адреса-порта источника и приемника. Это единственный способ выделить трафик от конкретного источника к конкретному получателю.
- 3) **Class-map** - дополнительный класс, который предоставляет дополнительные критерии фильтрации и может быть вложен в другой класс.
- 4) **None** - определяет, что любой трафик, который не соответствует указанному сервису или протоколу или листу доступа будет выбран в данном class-map.

Используя эти “фильтры” можно очень точно выбрать интересующий трафик, который мы можем впоследствии инспектировать. Пример создания class-map:

```
Router(config)#class-map type inspect match-any UserServices
Router(config-cmap)#match protocol http
Router(config-cmap)#match protocol https
Router(config-cmap)#match protocol ssh
Router(config-cmap)#match protocol ftp
Router(config-cmap)#match protocol smtp
```

В данном случае для class-map задано имя UserServices. Стоит обратить внимание на параметр **match-any**, который соответствует логическому ИЛИ. Т.е. трафик будет попадать под этот class-map если он будет соответствовать хотя бы одному протоколу (http, https, ssh, ftp или smtp).

Также существует параметр **match-all**, который соответствует логическому И, а значит трафик должен удовлетворять всем условиям, чтобы попасть под class-map. Данный параметр чаще всего используется совместно с access-group, когда трафик нужно отфильтровать не только по протоколу, но и по адресам источника и получателя. Рассмотрим небольшой пример. Предположим, что вы хотите разрешить вышеуказанные протоколы только для одной сети - 192.168.1.0/24. Для начала создадим access-list:

```
Router(config)#ip access-list extended Internet-Access
Router(config-ext-nacl)#permit ip 192.168.1.0 0.0.0.255 any
```

Теперь создадим новый class-map:

```
Router(config)#class-map type inspect match-all For-Inet-Access
Router(config-cmap)#match access-group name Internet-Access
Router(config-cmap)#match class-map UserServices
```

Как можно заметить, в данном случае указано два параметра, это **access-list** и ранее созданный **класс**. Таким образом трафик попадет под новый class-map если одновременно будут выполняться два условия:

- 1) Источником трафика является сеть 192.168.1.0/24
- 2) Используется любой протокол из: http, https, ssh, ftp, smtp.

*Выделите интересующий трафик с помощью **class-map**.*

6.2.5 Policy-map

Классифицировав трафик можно применять к нему какие-либо действия. Здесь у нас три варианта:

- 1) **Drop** - трафик просто отбрасывается (стоит понимать разницу между drop и reject). Стоит отметить, что каждая policy-map имеет по умолчанию class-default, для которого сконфигурировано действие “drop” (аналогично строке deny any any в любом списке доступа).
- 2) **Pass** – трафик пропускается без какой-либо инспекции. Это действие позволяет маршрутизатору пересылать трафик из одной зоны в другую, при этом не отслеживая состояние соединений или сессий. Это действие разрешает прохождение трафика только в одном направлении. Чтобы трафик прошел в обратном направлении, должна быть соответствующая политика для него. Это действие полезно для таких протоколов, как IPSec ESP, IPSec AH, ISAKMP и других “безопасных” протоколов с предсказуемым поведением.
- 3) **Inspect** - динамическая инспекция трафика, который проходит от зоны источника к зоне назначения. Обратный трафик автоматически разрешается даже для сложных протоколов, таких как H323. Т.е. используется технология Stateful Inspection.

Настройка политик выглядит следующим образом:

```
Router(config)#policy-map type inspect Internet-Policy
Router(config-pmap)#class type inspect For-Inet-Access
Router(config-pmap-c)#inspect
```

В данном примере для policy-map мы задали имя Internet-Policy. Политика применяется к трафику описанному в For-Inet-Access (class-map), а в качестве действия выбрано **inspect**. При необходимости можно заменить inspect на **pass** или **drop**. После создания политики к ней автоматически добавляется **class-default** с запрещающим действием. В конфигурации это выглядит следующим образом:

```
policy-map type inspect Internet-Policy
class type inspect For-Inet-Access
inspect
class class-default
drop
```

Для действия drop можно (я бы сказал нужно) добавлять команду log, чтобы логировать весь отбрасываемый трафик (**drop log**).

*С помощью **policy-map** определите действия над трафиком (**drop, pass, inspect**).*

6.2.6 Service-policy

Сформировав политику (policy-map) нам необходимо ее применить. Делается это с помощью **service-policy**. Сама настройка элементарна. Рассмотрим пример для трафика из внутренней сети в сеть Интернет:

```
Router(config)#zone-pair security Inside_Outside source Inside destination Outside
Router(config-sec-zone-pair)#service-policy type inspect Internet-Policy
```

Т.е. мы выбираем zone-pair и применяем к ней соответствующую политику. Без этого действия по умолчанию весь трафик будет запрещен.

*С помощью **service-policy** примените **policy-map** ко всем **zone-pair**.*

6.2.7 Интерфейсы

Последним шагом в настройке ZBF является включение интерфейсов в соответствующие зоны. Данный пункт является последним не просто так, ведь если это сделать в самом начале, то в течении всей последующей настройки трафик блокируется.

Определение интерфейса в зону производится следующим образом:

```
Router(config)#interface Ethernet0/0  
Router(config-if)#zone-member security Inside
```

Точно также настраиваются все остальные интерфейсы. Как уже говорилось ранее, в одну зону может входить сразу несколько интерфейсов (как физических, так и логических) и трафик между ними по умолчанию разрешен.

Включите интерфейсы в соответствующие зоны.

6.2.8 DoS

DoS (Denial of Service) - атака рассчитанная на отказ в обслуживании. Весьма часто, хакеру гораздо проще “заддосить” компьютер или сервер жертвы, чем взламывать его каким-то более хитрым способом. Существует огромное количество разновидностей DoS-а (DDoS) и способов их применения. Более того, в последнее время все большую популярность приобретает направление “**DDoS as a service**”. Это сервис для заказа DDoS атак. Все что вам нужно - заплатить несколько долларов и указать жертву, все остальное за вас сделают профессионалы.

Как вы уже поняли, DoS это очень обширная тема и естественно мы не можем подробно рассмотреть ее в рамках данной книги. Однако, хотелось бы упомянуть о возможности борьбы с некоторыми видами DoS атак с помощью Zone-Based Firewall. ZBF отлично справляется с атаками типа **SYN-флуд**, когда хакер посылает множество SYN-запросов, на каждый из которых компьютер/сервер жертвы (например Web-сервер) вынужден выделять системные ресурсы. Исчерпав ресурсы сервис становится недоступен. Примечательно, что для данной атаки не нужно тысячи компьютеров или “широкий” канал связи. Подобную атаку можно осуществить с одного компьютера подключенного через 3g модем. ZBF позволяет бороться с этим путем ограничения количества сессий.

Мы не будем рассматривать пример этого функционала, т.к. для этого потребуется большой объем вводной информации, которая выходит за рамки нашей книги. Однако если вам действительно интересна данная тема, то вы сможете найти большое количество информации по поисковому запросу типа: “**Zone-Based Firewall DoS**”. Просто не забывайте о такой возможности.

*При необходимости используйте функционал **ZBF** для борьбы с **DoS** - атаками.*

Повторюсь, что ZBF не является профессиональным средством защиты от DoS-атак и не стоит на него рассчитывать во время серьезной атаки.

6.3 DMZ

DMZ (Demilitarized Zone) - сегмент публичных серверов, т.е. ресурсов, которые должны быть доступны из сети Интернет. Главная цель DMZ - отделить публичные сервера от корпоративной сети.

Публичные сервера компании являются одним из самых уязвимых мест любой сети (хотя нерадивые пользователи гораздо опаснее). Если рассматривать локальную сеть, то она находится в относительной безопасности, т.к. “прячется” за NAT-ом и недоступна из внешней сети. В то же время, публичные сервера как правило видны абсолютно всем (в том числе и хакерам). А доступность серверов это уже большая угроза при неграмотных настройках и безответственном администрировании.

Мы уже частично затронули тему защиты DMZ описывая процесс настройки Zone-Based Firewall. В следующих параграфах мы постараемся рассмотреть детали и самое главное причины, по которым нам следует уделить особое внимание защите DMZ.

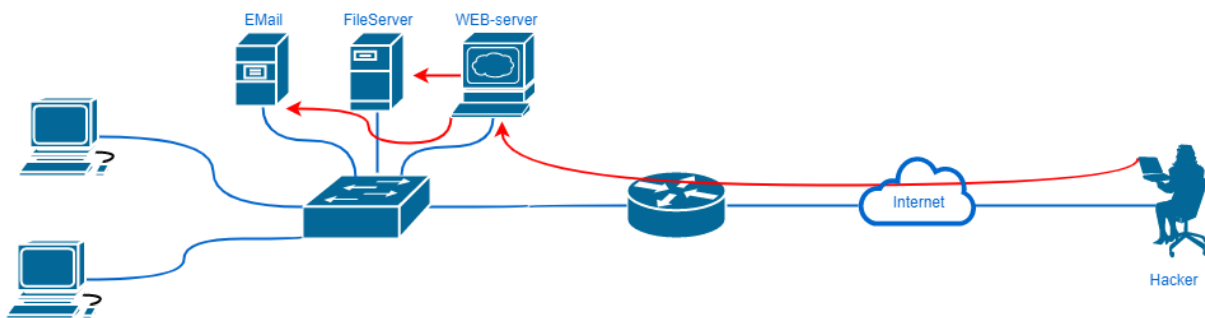
***DMZ** - один из самых уязвимых сегментов сети.*

6.3.1 Зачем?

Один из самых распространенных вопросов среди новичков: “Зачем публичные сервера выводить в отдельный сегмент?”. Действительно, ведь есть такая замечательная вещь как “проброс портов”. Оставляем сервер в локальной сети, настраиваем Static NAT, в результате при подключении на внешний ip-адрес роутера, указав определенный порт, попадаем на наш сервер из сети Интернет. Пример: Подключаемся на **210.210.1.2:8080**, запрос перенаправляется на **192.168.1.100:80**. Многие администраторы используют именно этот метод. В чем же его недостаток?

Как было сказано ранее, доступность какого-либо сервиса из сети Интернет это уже большая угроза. Если хакер имеет прямой доступ к системе (пусть это даже всего лишь один порт), то вероятность “взлома” возрастает многократно. Однако это не самое страшное, ведь на общий доступ публикуются ресурсы, которые должны быть доступны всем (или ограниченном кругу лиц), а значит скорее всего они не содержат критически важную информацию (хотя бывают и исключения). Гораздо страшнее, что взломав открытый ресурс, хакер может получить доступ к закрытым ресурсам компании. Чаще всего злоумышленники используют публичные ресурсы именно как плацдарм для дальнейшей атаки.

Вы можете опубликовать в Интернете обычный сайт с публичной информацией о вашей компании. Взломав этот ресурс хакер окажется уже за вашим межсетевым экраном или роутером, а это значительно упрощает все дальнейшие махинации.



Используя взломанный веб-сервер злоумышленник может попытаться получить доступ например к файловому или почтовому серверу. При этом ему ничего не будет мешать, т.к. он уже находится в их сегменте.

Именно поэтому настоятельно рекомендуется выносить публичные сервера в отдельный сегмент. Это может быть VLAN или же отдельный физический интерфейс маршрутизатора. Главное, чтобы между локальной сетью и DMZ всегда находилось L3 устройство, причем желательно, чтобы это устройство поддерживало технологию stateful inspection (в нашем случае ZBF). Т.е. не стоит организовывать DMZ на L3 коммутаторе.

*Не используйте “проброс портов” для публикации ресурсов в сеть Интернет.
 Используйте DMZ.*

Таким образом вы сможете защитить свою локальную сеть и важные данные, даже при взломе публичного сервера (обычно это веб-сайт).

Стоит отметить, что гораздо удобнее организовывать DMZ с использованием “белых” (т.е. публичных) ip-адресов. В этом случае вам не придется дополнительно настраивать Static NAT. Однако, если такая возможность отсутствует, то можно обойтись “серой” сетью с последующим “пробросом” портов.

Если же вы не можете выделить DMZ, а некоторым сотрудникам требуется организовать доступ к внутренним ресурсам, то используйте Remote Access VPN. Так вы откроете доступ лишь для разрешенных пользователей.

6.3.2 Лишние сервисы

Увидев публичный сервер в DMZ, первое, что сделает злоумышленник - просканирует открытые порты. Для этого существует огромное количество бесплатных инструментов, таких как **Nmap**. Пример просканированного узла:

```
Starting Nmap 5.00 ( http://nmap.org ) at 2012-11-27 03:30 IST
Interesting ports on 192.168.1.1:
PORT      STATE SERVICE
21/tcp    closed ftp
22/tcp    open  ssh
23/tcp    closed telnet
25/tcp    closed smtp
80/tcp    open  http
110/tcp   closed pop3
139/tcp   closed netbios-ssn
443/tcp   closed https
445/tcp   closed microsoft-ds
3389/tcp  closed ms-term-serv
MAC Address: BC:AE:C5:C3:16:93 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds
```

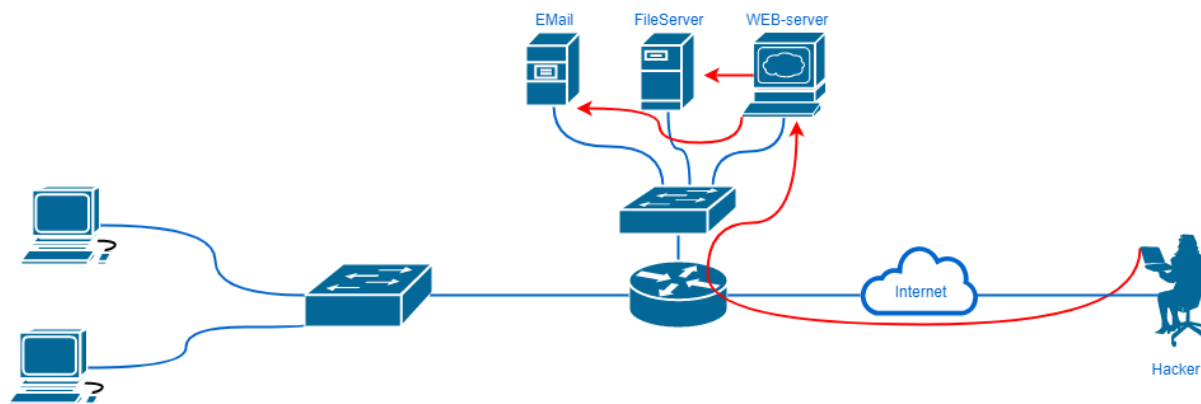
Определив открытые порты, хакер может сделать вывод о запущенных сервисах на данном сервере. В свою очередь эта информация позволяет ему начать поиск возможных уязвимостей, которые появляются весьма быстро. При этом нет гарантии, что вы не пропустите очередное критическое обновление и ваш сервер не имеет “дыр” на текущий момент.

Логично, что в связи с описанной проблемой, стоит уделить особое внимание актуальности обновлений безопасности для всех серверов. Мы это упоминали и ранее. Однако еще важнее - не открывать лишних портов! Чем меньше портов “светится” в Интернете, тем лучше. В случае использования ZBF мы можем указывать конкретные протоколы и сервисы, такие как http, https, ssh, smtp и так далее. Не забывайте пользоваться этой возможностью и открывать только действительно необходимые сервисы.

Закройте все лишние сервисы в DMZ.

6.3.3 Private VLAN

Мы уже обсудили возможность использования взломанного публичного сервера в качестве площадки для дальнейших атак на локальные сети. Данная проблема решается путем создания DMZ. Однако существует еще одна опасность. Предположим, что в DMZ находится сразу несколько сервисов: Web-server, Email-сервер, FTP сервер.



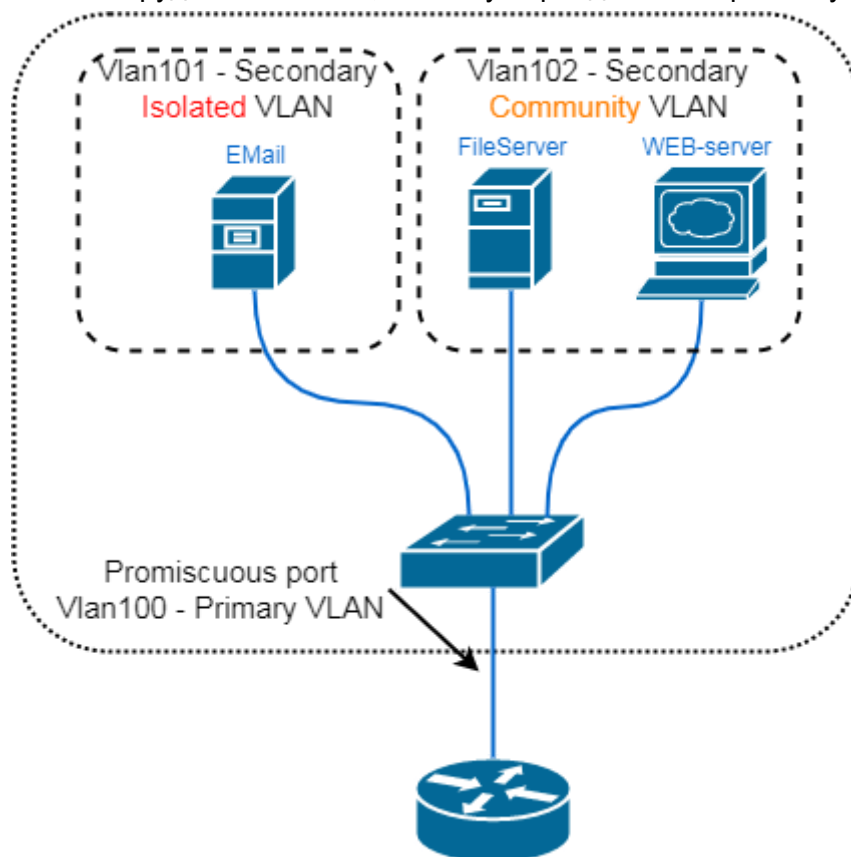
Как вы понимаете, все эти сервера находятся в одном сегменте. А это значит, что если злоумышленник “взломает” один из серверов, то он получит возможность беспрепятственно атаковать остальные хосты данного сегмента, ведь он будет уже за межсетевым экраном. Таким образом, защищенность всего сегмента будет определяться самым слабым (с точки зрения безопасности) звеном.

Для решения данной проблемы, первое, что может прийти в голову - сегментирование. Однако, это далеко не лучший вариант. Помещая каждый публичный сервер в отдельный сегмент с выделенной подсетью, мы существенно усложняем конфигурацию и администрирование. Да и где взять столько “белых” подсетей? Гораздо более логичным и “красивым” решением является использование технологии **Private VLAN**.

Private VLAN - технология, которая позволяет запретить общение между хостами, даже если они находятся в одном VLAN-е. Достигается это за счет создания **Primary** и **Secondary** VLAN-ов. Primary - самый обычный VLAN, к которым мы привыкли. Secondary VLAN находится как бы внутри Primary и может работать в двух режимах:

- 1) **isolated** – это полностью изолированные порты. Даже если несколько хостов будут находиться в одном isolated VLAN-е, то они не смогут общаться.
- 2) **community** – хосты находящиеся в рамках одного community VLAN-а могут общаться друг с другом, но не могут общаться с другими private VLAN-ами.

Эти два типа Secondary VLAN-а объединяются в один Primary, который как правило находится в режиме **promiscuous** и может передавать информацию для всех. Словами это весьма трудно объяснить, поэтому перейдем к конкретному примеру:



Если в сегменте DMZ находится сразу несколько серверов, то логично предположить наличие коммутатора. У нас имеется три сервера и мы хотим полностью изолировать EMail сервер от остальных сервисов данного сегмента. При этом необходимо оставить возможность для "общения" между Web- и File-серверами. Также мы не хотим дробить сеть на дополнительные сегменты. Для начала нужно включить нужный режим VTP:

```
Switch(config)#vtp mode transparent
```

Определяем Primary VLAN:

```
switch(config)#vlan 100
switch(config-vlan)#private-vlan primary
switch(config-vlan)#exit
```

Теперь создадим Secondary VLAN в режиме isolated (для EMail-сервера):

```
switch(config)#vlan 101
switch(config-vlan)#private-vlan isolated
switch(config-vlan)#exit
```

Для Web- и File-серверов также создадим Secondary VLAN но уже в режиме community, чтобы они смогли общаться друг с другом:

```
switch(config)#vlan 102
switch(config-vlan)#private-vlan community
switch(config-vlan)#exit
```

Определим, что Secondary VLAN-ы принадлежат нашему Primary VLAN-у:

```
switch(config)#vlan 100
switch(config-vlan)#private-vlan association 101-102
switch(config-vlan)#exit
```

Теперь настроим мапинг Secondary и Primary VLAN-ов на физических портах:

```
switch(config)#interface Fa0/2      /порт EMail-сервера
switch(config-if)#switchport mode private-vlan host
switch(config-if)#switchport private-vlan host-association 100 101
switch(config-if)#exit
switch(config)#interface Fa0/3      /порт Web-сервера
switch(config-if)#switchport mode private-vlan host
switch(config-if)#switchport private-vlan host-association 100 102
switch(config-if)#exit
switch(config)#interface Fa0/4      /порт File-сервера
switch(config-if)#switchport mode private-vlan host
switch(config-if)#switchport private-vlan host-association 100 102
switch(config-if)#exit
switch(config)#interface Fa0/1      /порт Маршрутизатора (uplink)
switch(config-if)#switchport mode private-vlan promiscuous
switch(config-if)#switchport private-vlan host-association 100 101-102
switch(config-if)#exit
```

Таким образом мы сможем изолировать трафик серверов, имеющих ip-адреса из одной подсети и находящихся в одном логическом сегменте. Это существенно повысит безопасность DMZ.

*По возможности используйте технологию **Private VLAN** в DMZ.*

Здесь стоит сразу упомянуть, что данная технология доступна только на L3-коммутаторах, таких как 3560. Далеко не каждая организация может поставить такой коммутатор для сегмента DMZ. В качестве альтернативы можно рассмотреть более простую технологию для L2-коммутаторов - switchport [protected](#). Мы же ограничимся описанием только private VLAN, как более перспективной технологии.

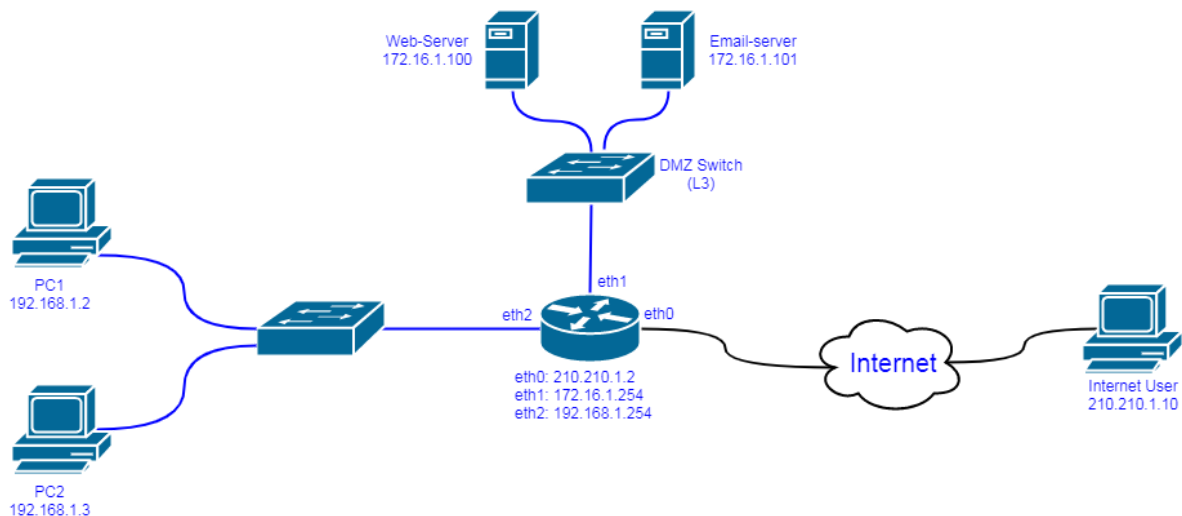
6.4 Чеклист №6

Подошла к концу заключительная глава. Ниже представлен краткий чеклист по защите периметра сети:

- 1) *Минимизируйте разрешенный трафик из сети Интернет.*
- 2) *Обязательно используйте технологии **Stateful Inspection** на периметре сети.*
- 3) *Убедитесь, что прошивка маршрутизатора поддерживает **Zone-Based Firewall**.*
- 4) *Проведите аудит с помощью **NetFlow** и составьте матрицу доступа для периметра.*
- 5) *Определите зоны безопасности на устройстве периметра.*
- 6) *Определите все возможные пары зон, между которыми должен “ходить” трафик.*
- 7) *Выделите интересующий трафик с помощью **class-map**.*
- 8) *С помощью **policy-map** определите действия над трафиком (**drop, pass, inspect**).*
- 9) *С помощью **service-policy** примените **policy-map** ко всем **zone-pair**.*
- 10) *Включите интерфейсы в соответствующие зоны.*
- 11) *При необходимости используйте функционал ZBF для борьбы с **DoS - атаками**.*
- 12) ***DMZ** - один из самых уязвимых сегментов сети.*
- 13) *Не используйте “проброс портов” для публикации ресурсов в сеть Интернет. Используйте DMZ.*
- 14) *Закройте все лишние сервисы в DMZ.*
- 15) *По возможности используйте технологию **Private VLAN** в DMZ.*

6.5 Пример конфигурации

Рассмотрим последний пример конфигурации. Ниже представлена классическая схема сети:



(С вашего позволения я опишу в данном примере только настройку Zone-Based Firewall.) Настройки интерфейсов маршрутизатора изображены на картинке. Для локальной сети настроен NAT. В DMZ находится два сервера (Web-Server, EMail-server), которые имеют “серые” IP-адреса, поэтому для доступа извне используется Static NAT (проброс портов). Т.е. перед началом настройки ZBF у вас должна быть примерно следующая конфигурация:

```
interface Ethernet0/0
description Outside
ip address 210.210.1.2 255.255.255.0
ip nat outside
ip virtual-reassembly
!
interface Ethernet0/1
description DMZ
ip address 172.16.1.254 255.255.255.0
ip nat inside
ip virtual-reassembly
!
interface Ethernet0/2
description Inside
ip address 192.168.1.254 255.255.255.0
ip nat inside
ip virtual-reassembly

ip access-list standard ForNAT
permit 192.168.1.0 0.0.0.255
permit 172.16.1.0 0.0.0.255

ip nat inside source list ForNAT interface Ethernet0/0 overload
ip nat inside source static tcp 172.16.1.100 80 210.210.1.2 80 extendable
ip nat inside source static tcp 172.16.1.100 25 210.210.1.2 25 extendable
```

Убедившись, что при такой конфигурации у пользователей есть доступ в интернет, а публичные сервера доступны для удаленных пользователей, можно приступить к настройке Zone-Based Firewall. Как было описано ранее, делается это в несколько шагов:

- 1) Как можно заметить, в данной схеме всего три зоны: **Inside**, **Outside** и **DMZ**. Названия зон естественно могут быть другими. Настройка зон:

```
Router(config)#zone security Outside
Router(config-sec-zone)#exit
Router(config)#zone security Inside
Router(config-sec-zone)#exit
```

```
Router(config)#zone security DMZ
Router(config-sec-zone)#exit
```

- 2) Для простоты предположим, что трафик будет ходить всего в трех направлениях:

```
Router(config)#zone-pair security Inside_Outside source Inside destination Outside
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Inside_DMZ source Inside destination DMZ
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Outside_DMZ source Outside destination DMZ
Router(config-sec-zone-pair)#exit
```

- 3) Теперь необходимо определить фильтры трафика, т.е. **class-map**. Здесь необходимо создать несколько классов.

Для начала нам необходимо сформировать “базовые” классы, которые мы сможем затем включать в другие class-мапы. Это как детали конструктора из которых собираются более сложные вещи. Определим трафик “обычных” пользователей:

```
Router(config)#class-map type inspect match-any UserServices
Router(config-cmap)#match protocol http
Router(config-cmap)#match protocol https
Router(config-cmap)#match protocol ftp
Router(config-cmap)#match protocol smtp
```

Теперь определим сервисы, которые необходимы для администраторов:

```
Router(config)#class-map type inspect match-any AdminServices
Router(config-cmap)#match protocol ssh
Router(config-cmap)#match protocol telnet
```

С помощью access-list-а определим компьютер администратора (в нашем случае это **PC1**), который может пользоваться сервисами ssh и telnet:

```
Router(config)#ip access-list extended AdminAccess
Router(config-ext-nacl)#permit ip host 192.168.1.2 any
```

Опишем трафик пользователей из локальной сети в Интернет:

```
Router(config)#class-map type inspect match-any For-Int-Access
Router(config-cmap)#match class-map UserServices
```

Как видите, мы создали новый class-мап, в который включили уже существующий класс **UserServices**. В следующий раз, если пользователям понадобится дополнительный уровень доступа (например все те же сервисы ssh и telnet), то мы просто добавим еще один класс, который тоже станет частью конструктора. Теперь опишем трафик пользователей из локальной сети в DMZ:

```
Router(config)#class-map type inspect match-any User-DMZ-Access
Router(config-cmap)#match class-map UserServices
```

В данном случае сервисы такие же как и для Интернета. Трафик для компьютера администратора в DMZ требует дополнительных сервисов:

```
Router(config)#class-map type inspect match-all Admin-DMZ-Access
Router(config-cmap)#match access-group name AdminAccess
Router(config-cmap)#match class-map AdminServices
```

С помощью параметра **match-all** мы связываем два условия - трафик должен попадать под список доступа **AdminAccess** и соответствовать сервисам класс **AdminServices** (ssh, telnet).

Опишем трафик из сети Интернет к серверам DMZ. В нашем случае нужны всего два сервиса (http, smtp):

```
Router(config)#class-map type inspect match-any InternetServices
Router(config-cmap)#match protocol http
Router(config-cmap)#match protocol smtp
```

- 4) После того, как трафик описан, необходимо сформировать **policy-map**, чтобы выбрать действия над интересующим трафиком. Всего у нас будет три policy, в соответствии с направлениями трафика.

Policy-map для трафика из локальной сети в сеть Интернет:

```
Router(config)#policy-map type inspect Internet-Policy
Router(config-pmap)#class type inspect For-Inet-Access
Router(config-pmap-c)#inspect
Router(config-pmap-c)#exit
Router(config-pmap)#exit
```

Policy-map для трафика из локальной сети в DMZ:

```
Router(config)#policy-map type inspect DMZ-Policy
Router(config-pmap)#class type inspect User-DMZ-Access
Router(config-pmap-c)#inspect
Router(config-pmap-c)#exit
Router(config-pmap)#class type inspect Admin-DMZ-Access
Router(config-pmap-c)#inspect
Router(config-pmap-c)#exit
Router(config-pmap)#exit
```

Policy-map для трафика из сети Интернет в DMZ:

```
Router(config)#policy-map type inspect Internet-DMZ-Policy
Router(config-pmap)#class type inspect InternetServices
Router(config-pmap-c)#inspect
Router(config-pmap-c)#exit
Router(config-pmap)#exit
```

- 5) Теперь применяем политики:

```
Router(config)#zone-pair security Inside_Outside source Inside destination Outside
Router(config-sec-zone-pair)#service-policy type inspect Internet-Policy
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Inside_DMZ source Inside destination DMZ
Router(config-sec-zone-pair)#service-policy type inspect DMZ-Policy
Router(config-sec-zone-pair)#exit
Router(config)#zone-pair security Outside_DMZ source Outside destination DMZ
Router(config-sec-zone-pair)#service-policy type inspect Internet-DMZ-Policy
Router(config-sec-zone-pair)#exit
```

- 6) Последним шагом интерфейсы маршрутизатора помещаются в соответствующие зоны:

```
Router(config)#interface Ethernet0/0
Router(config-if)#zone-member security Outside
Router(config-if)#exit
Router(config)#interface Ethernet0/1
Router(config-if)#zone-member security DMZ
Router(config-if)#exit
Router(config)#interface Ethernet0/2
Router(config-if)#zone-member security Inside
```

Router(config-if)#exit

На этом настройка ZBF завершена. Естественно, что мы рассмотрели простейший случай, однако общий принцип должен быть понятен.

Заключение

После прочтения данной книги вы с большой долей вероятности обнаружите, что ваша текущая сеть нуждается в серьезных изменениях, которые потребуют немало усилий. Возможно на правку потребуется несколько выходных, т.к. в рабочее время недопустимы простои сети. От себя лишь могу посоветовать, что не стоит лениться и воспринимать данную работу как “проблему”. Относитесь к подобным задачам, как к отличным возможностям получить бесценный опыт.

При этом вы должны понимать, что защита современной сети не ограничивается описанными методами. Чтобы обеспечить комплексную информационную безопасность вы будете просто вынуждены использовать такие решения как Прокси-сервера, Поточные Антивирусы, Межсетевые экраны следующего поколения (NGFW), системы предотвращения вторжений (IPS), DLP системы, SIEM системы и т.д. Об этом и многом другом я вкратце рассказываю в предыдущей книге [“Архитектура корпоративных сетей”](#). Кроме того, в ближайшем будущем планируется книга по современным средствам защиты сети, где будут рассматриваться UTM-решения и Enterprise Firewall.

Что ж, вот и подошла к концу моя вторая книга. Какой она получилась, судить только вам, но я с большим удовлетворением пишу последние строки. Я постарался отразить в ней все мои знания и опыт по защите сетей без специализированных средств защиты. Это была отличная возможность структурировать имеющуюся информацию у меня в голове. Искренне надеюсь, что этот материал кому-нибудь действительно пригодится.

Спасибо, что прочитали до самого конца!

P.S. По всем возникающим вопросам вы можете обращаться по адресу - e.olkov@netskills.ru.

P.P.S. Еще раз хотел бы обратиться к читателям с просьбой не распространять эту книгу в сети Интернет. Автор потратил очень много сил на ее написание. Хоть и небольшой доход от продаж книги позволяет поддерживать и развивать бесплатный проект NetSkills, где и в дальнейшем будут публиковаться бесплатные видео курсы. Заранее благодарен за соблюдение авторского права!