Практикалық сабақ №10: Address Spoofing шабуылынан қорғау

То, что Интернет является агрессивной средой, уже давно ни для кого не является новостью. Стоимость информации может быть настолько высока, что с лихвой окупает любые затраты на ее получение. Кроме того, обострение конкуренции подчас приводит к выбору весьма агрессивных методов борьбы с «коллегами по цеху», одним из которых является вывод из строя или временная недоступность сетевого оборудования конкурента. Ну и с каждым днем все острее становится проблема сетевого хулиганства. Умение защищать сервер и локальную сеть предприятия от подобных атак становится обязательным требованием к квалификации системного администратора даже на небольших предприятиях, на которых самая секретная информация – почтовые ящики сотрудников.

Известно, что любая грамотная атака начинается с разведки – сбора всей доступной информации об объекте нападения. Цель данной статьи – познакомить читателя с основными методами подобной разведки и способами защиты от них с помощью пакетного фильтра ipfw. Попутно нам придется подробно рассмотреть базовые протоколы передачи данных (IP, TCP, UDP) и программу tcpdump, позволяющую осуществлять контроль за проходящими через машину пакетами.

Стек протоколов TCP/IP

Протоколы TCP/IP являются базовыми в сети Интернет. В эталонной модели OSI они занимают верхние уровни, начиная с сетевого (протоколы ARP и RARP также частично выполняют функции канального уровня). Поскольку наша задача – защита с помощью брандмауэра ipfw, работающего на сетевом и транспортном уровнях, то подробно рассмотрим интернет-протоколы этих двух уровней.

Таблица 1

Уровень OSI	Протоколы
7. Приложений	HTTP, FTP, SMTP, SNMP, Telnet и др.
6. Представлений	
5. Сессионный	
4. Транспортный	TCP, UDP
3. Сетевой	IP, ICMP, протоколы маршрутизации
2. Канальный	--
1. Физический	--

Протоколы сетевого уровня

Основным интернет-протоколом сетевого уровня является протокол IP. В его задачу входит маршрутизация пакетов в соответствии с IP-адресами, то есть определение маршрута следования пакета от источника к приемнику и передача его по требуемому адресу, а также фрагментация и сборка пакетов верхних уровней. Основной формат заголовка IP-пакета представлен на рисунке:

0:Version	HdrLen	1:Type of service	2:Packet Length
4:Fragmentation			
8:Time to live (TTL)		9:Protocol	10:Header Checksum
12:Source IP-address			
16:Destination IP-address			

Первый байт делится между номером версии протокола (4 для IPv4) и длиной IP-заголовка HdrLen (как правило 20 байт). Следующий байт задает тип обслуживания пакета (TOS), далее два байта – общая длина пакета. Следующие восемь байт содержат информацию о фрагментации пакета, необходимую для его последующей сборки. Среди прочего здесь располагается флаг DF, установка которого запрещает фрагментацию пакета. Девятый и десятый байты содержат информацию соответственно о времени жизни пакета (TTL) и протоколе

(Protocol) верхнего уровня, которому пакет должен быть передан для дальнейшей обработки; затем два байта – контрольная сумма. Замыкают заголовок IP-адреса источника и приемника, занимающие по четыре байта. В общем случае далее могут следовать опциональные поля (их количество определяется длиной заголовка пакета). Затем идут собственно пользовательские данные. Под пользователем в данном случае понимается протокол более высокого уровня (TCP или UDP).

Наиболее важной информацией, содержащейся в IP-заголовке, для нас является время жизни пакета и адреса источника и приемника. Заметим, что информации о портах в IP-заголовке нет, поскольку данная информация используется на транспортном уровне и не требуется для маршрутизации пакета.

Помимо IP-протокола на сетевом уровне располагается также протокол ICMP – Internet Control Message Protocol, который используется для обмена служебной информацией между хостами. Наиболее известный и наглядный пример использования этого протокола – команда ping. Заголовок ICMP-пакета во многом идентичен IP-заголовку.

Протоколы транспортного уровня

Транспортный уровень представлен протоколами TCP и UDP. Общий формат заголовка TCP-пакета:

0: Source Port		2: Destination Port
4: Sequence number		
8: Acknoledgement number		
12: HdrLen	13: Flags	14: Window
16: Checksum		18: Urgent Pointer

Первые четыре байта содержат информацию о портах источника и получателя (по два байта каждый). Номер последовательности (4 байта) используется для нумерации передаваемых байтов (позволяет контролировать порядок получения и сборки). Следующие четыре байта (номер подтверждения) содержат номер последовательности следующего (ожидаемого) байта и устанавливаются при подтверждении получения предыдущего пакета. Далее следует четыре бита длины заголовка и четыре зарезервированных для дальнейшего использования бита. Чрезвычайно важный 14-й байт (флаги TCP-заголовка) подробно рассмотрен в следующем абзаце. Далее по два байта занимает информация об окне (размере буфера) приема (Window), контрольной сумме и указателе на первый байт данных, помеченный на первоочередную обработку (Urgent).

Ниже представлен формат 14-го байта, содержащего флаги TCP-пакета:

--	--	URG	ACK	PSH	RST	SYN	FIN

- **FIN** – флаг «мягкого» завершения соединения. При получении пакета с этим флагом удаленная сторона передает информацию, накопленную в буфере, и завершает сеанс;
- **SYN** – флаг «синхронизации» используется при запросе на установление соединения. Так, при желании установить связь система выставляет пакет с флагом SYN. Удаленная система отвечает на него пакетом с установленными битами SYN и ACK (см. ниже). В ответ на этот пакет высылается пакет подтверждения с флагом ACK, после чего TCP-соединение считается установленным;
- **RST** – сигнал «сброса» (жесткий разрыв соединения). При получении пакета с таким битом удаленная система должна прекратить сеанс связи немедленно;
- **PSH** – команда на принудительную внеочередную передачу информации, накопленной в буфере (в нормальном режиме TCP-пакеты отсылаются не сразу, а предварительно накапливаются в буфере передачи и отсылаются только при достижении определенного размера буфера);
- **ACK** – флаг «подтверждение приема». Данный флаг должен выставляться в ответ на безошибочный прием любого пакета;
- **URG** – пакет содержит данные, имеющие высокий приоритет (которые должны обрабатываться в первую очередь).

Наиболее важной с точки зрения фильтрации информацией TCP-заголовка являются флаги и адреса портов. IP-адреса в данном заголовке не передаются, поскольку за маршрутизацию отвечает сетевой уровень, обслуживаемый протоколом IP.

Формат UDP-заголовка намного проще:

0: Source Port	2: Destination Port
4: Length	6: Checksum

То есть в данном случае передаются номера портов источника и приемника, длина пакета и контрольная сумма заголовка, после чего следуют пользовательские данные.

Основные способы сканирования портов

Для того чтобы получить информацию об атакуемой машине или сети, чаще всего используется сканирование портов. Этот метод заключается в том, что последовательно отсылаются пакеты, предназначенные различным портам удаленной машины, и на основании полученных ответов делается вывод о том, открыт данный порт (то есть работает ли программа, обслуживающая пакеты, приходящие на него) или закрыт. Также по характеру реакции «жертвы» на нестандартные пакеты довольно часто удается определить операционную систему, работающую на сканируемой машине, и версии обслуживающих соответствующие порты программ.

Одной из наиболее распространенных в среде UNIX утилит для определения доступных портов является программа nmap. Для FreeBSD она может быть установлена из коллекции портов: /usr/ports/security/nmap.

Упрощенно формат ее запуска следующий:

```
# nmap [options] hosts
```

Параметр hosts задает список сканируемых хостов через запятую или диапазон сканируемых адресов через дефис. Опции задают метод сканирования и могут иметь следующие значения:

- **-sT** – обычное сканирование, когда соединение с каждым портом устанавливается полностью, а затем разрывается;
- **-sS** – полусоединение, когда связь разрывается после ответа SYN+ACK удаленной машины;
- **-sF** – сканирование пакетами с установленным флагом FIN. Поскольку соединение не установлено, то удаленная система обрабатывает эту ситуацию как ошибочную, причем для различных операционных систем реакция может быть различной, что в ряде случаев позволяет также определить ОС, работающую на сканируемой машине;
- **-sX** – в сканирующих пакетах устанавливаются все флаги;
- **-sN** – в сканирующих пакетах не установлен ни один флаг. Поскольку обе эти ситуации не предусмотрены протоколом, то по реакции на нее удаленной системы можно попытаться определить тип ОС и версии запущенных программ;
- **-sP** – обычный ping, то есть порты не сканируются, а просто проверяется доступность всех хостов, перечисленных в параметрах команды, с помощью пакетов «Echo Request» протокола ICMP;
- **-sU** – сканирование UDP-портов.

Еще два полезных флага: -O, который инициирует попытку определить операционную систему на удаленной машине, и –I, по которому nmap пытается определить пользователя, с правами которого запущена служба, обслуживающая сканируемый порт.

Помимо nmap существует множество программ, в той или иной мере решающих задачу определения открытых портов на удаленной машине. Одной их таких для среды Windows является XSpider, разрабатываемый компанией Positive Technologies.

Программа tcpdump

В состав большинства операционных систем семейства Unix входит очень полезная утилита tcpdump, которая позволяет системному администратору просматривать и анализировать пакеты, проходящие через машину, на которой она запущена. Для работы с ней вы должны иметь права root.

Утилита имеет следующие ключи запуска:

```
tcpdump    [ -adeflnNOpqStvx ] [ -c count ] [ -F file ]

           [ -i interface ] [ -r file ] [ -s snaplen ]

           [ -T type ] [ -w file ] [ expression ]
```

Ознакомиться со всеми ключами можно на странице справочного руководства man tcpdump(1). Здесь мы рассмотрим только наиболее полезные параметры.

Запущенная без дополнительных параметров, tcpdump будет выводить на экран заголовки всех пакетов, проходящих через все активные интерфейсы системы. Список прослушиваемых интерфейсов будет отображен в первой строке:

```
# tcpdump
```

```
tcpdump:                        listening                       on                        ed1
15:51:06.704430 imns.server.ru.esl-lm > remote.host.ru.ftp-data: . ack 3437419348 win 8576 (DF)
15:51:06.768457 gw-sovm.chtts.ru.ftp-data > imns.host.ru.esl-lm: . 1:537(536) ack 0 win 65535
(DF)
…
```

В общем случае формат выводимой информации следующий:

```
timestamp src.port > dst.port: flags first:last(bytes) ack win urg options
```

Timestamp – время прохождения пакета с точностью до микросекунд. Далее следуют адреса и порты источника и приемника (адрес и порт разделены точкой). И адрес, и порт по возможности представляются в символьном виде. Если определить его не удается, печатается IP-адрес и номер порта. После двоеточия следует перечисление установленных флагов («S» – SYN, «F» – FIN, «P» – PSH, «R» – RST, «.» – ни один из этих флагов не установлен). Следующий необязательный блок задает начальный (first) и конечный (last) номера последовательности пакетов и число байт (bytes) пользовательских данных в ней. Конечный номер (last) в последовательность не включается. Запись «ack» включается в строку, если пакет содержит флаг ACK. «Win» указывает размер окна приема, «urg» устанавливается, если пакет имеет данные с высоким приоритетом и должен обрабатываться в первую очередь (установлен флаг URG). Секция «options» может содержать дополнительную информацию о пакете, заключаемую в угловые скобки.

В качестве примера рассмотрим, как выглядит в tcpdump запрос к веб-серверу 1.2.3.4 со стороны хоста host.ru. Для сокращения записи временные штампы опущены (этого можно достичь с помощью параметра –t):

```
# tcpdump –t
```

```
tcpdump:                    listening                    on                    ed1
host.ru.2200    >    1.2.3.4.http:    S    35208225:35208225(0)    win    8192    (DF)
1.2.3.4.http > host.ru.2200: S 2830515394:2830515394(0) ack 35208226 win 17520 (DF)
host.ru.2200     >     1.2.3.4.http:      .     ack     1     win     8760     (DF)
host.ru.2200    >    1.2.3.4.http:    P    1:501(500)    ack    1    win    8760    (DF)
1.2.3.4.http     >     host.ru.2200:      .     ack     501     win     17520     (DF)
1.2.3.4.http    >    host.ru.2200:    .    1:1461(1460)    ack    501    win    17520    (DF)
1.2.3.4.http    >    host.ru.2200:    .    1461:2921(1460)    ack    501    win    17520    (DF)
1.2.3.4.http    >    host.ru.2200:    .    2921:4381(1460)    ack    501    win    17520    (DF)
1.2.3.4.http    >    host.ru.2200:    P    4381:4453(72)    ack    501    win    17520    (DF)
host.ru.2200     >     1.2.3.4.http:      .     ack     2921     win     8760     (DF)
1.2.3.4.http    >    host.ru.2200:    P    4453:5295(842)    ack    501    win    17520    (DF)
host.ru.ats    >    1.2.3.4.http:    S    35208497:35208497(0)    win    8192    (DF)
host.ru.2200     >     1.2.3.4.http:      .     ack     4381     win     8760     (DF)
1.2.3.4.http > host.ru.ats: S 2830609828:2830609828(0) ack 35208498 win 17520 (DF)
host.ru.ats     >     1.2.3.4.http:      .     ack     1     win     8760     (DF)
host.ru.ats    >    1.2.3.4.http:    P    1:619(618)    ack    1    win    8760    (DF)
1.2.3.4.http    >    host.ru.ats:    P    1:217(216)    ack    619    win    17520    (DF)
host.ru.ats     >     1.2.3.4.http:      .     ack     217     win     8544     (DF)
1.2.3.4.http    >    host.ru.2200:    P    4381:5295(914)    ack    501    win    17520    (DF)
host.ru.2200    >    1.2.3.4.http:    P    501:1077(576)    ack    5295    win    7846    (DF)
1.2.3.4.http     >     host.ru.2200:      .     ack     1077     win     17520     (DF)
```

```
1.2.3.4.http > host.ru.2200: P 5295:6039(744) ack 1077 win 17520 (DF)
host.ru.2200 > 1.2.3.4.http: . ack 6039 win 8760 (DF)
……
```

В первой строке хост host.ru со своего порта 2200 отправляет на порт http хоста 1.2.3.4 запрос на установку соединения (флаг SYN). Начальный и конечный номера последовательности совпадают, поскольку пользовательская информация при этом не передается. Размер буфера приема – 8192 байта. В угловых скобках передается дополнительная информация, в том числе параметр mss (max segment size), ограничивающий количество пользовательской информации в одном пакете. Признак (DF) сообщает о том, что фрагментация данного пакета запрещена. Хост 1.2.3.4 отвечает пакетом с установленным флагом SYN и флагом подтверждения ACK, вместе с которым передается и номер подтверждения (номер последовательности, который хост 1.2.3.4 будет ожидать в следующем пакете). В третьей строке host.ru отвечает пакетом с флагом ACK (далее нумерация последовательностей относительная, то есть номер 1 означает, что ожидается первый пакет в данном сеансе связи). После этого соединение считается установленным.

В следующем пакете host.ru отправляет пользовательский запрос (500 байт) с установленным флагом очистки буфера PSH, а заодно подтверждает прием предыдущего пакета (установленный флаг ACK). В ответ хост 1.2.3.4 отсылает серию пакетов с запрошенной информацией (первые четыре по 1460 байт (действует ограничение mss, установленное ранее и запрещающее передавать в одном пакете больше 1460 байт пользовательской информации), в последнем – оставшиеся 72 байта). Далее host.ru отсылает подтверждения на полученные пакеты, одновременно открывая еще одно соединение, скорее всего для загрузки изображения, содержащегося на запрошенной странице.

Рассмотрим наиболее полезные параметры программы tcpdump:

- **-c №** – завершает работу после получения № пакетов;
- **-e** – выводит информацию, содержащуюся в заголовке канального уровня. Данная информация выводится между штампом времени и адресом источника и содержит, в частности, MAC-адреса источника и приемника;
- **-i interface** – позволяет прослушивать указанный интерфейс (по умолчанию прослушиваются все активные интерфейсы);
- **-n** – все адреса хостов и порты представляются в цифровом виде;
- **-N** – не печатать полное доменное имя. Например, если данная опция установлена, то адрес и порт «host.server.ru.http» будут представлены в виде «host.http»;
- **-q** – минимизирует выводимую информацию. Результат будет примерно таким:

 # tcpdump -c 5 –q

```
                    tcpdump: listening on ed1
10:30:20.447422 1.2.3.4.ssh > suprunov.host.ru.cadis-2: tcp 20 (DF) [tos 0x10]
 10:30:20.483689 imns.host.ru.1254 > gw-sovm.chtts.ru.ftp-data: tcp 0 (DF)
10:30:20.527295 gw-sovm.chtts.ru.ftp-data > imns.host.ru.1254: tcp 536 (DF)
10:30:20.599219 vipdosug2.masterhost.ru.http > d4.host.ru.4126: tcp 1460 (DF)
10:30:20.619398 suprunov.host.ru.cadis-2 > 1.2.3.4.ssh: tcp 0 (DF)
```

То есть указываются только адреса и порты источника и приемника, протокол верхнего уровня, размер переданной информации и некоторые дополнительные сведения. Использование этого параметра совместно с опцией «-t» (см. ниже) позволит еще больше сократить выводимую информацию, и если повезет, то строки даже смогут полностью поместиться на экране.

- **-S** – выводит абсолютные, а не относительные, номера последовательности для установленных соединений;
- **-t** – не выводит на экран штамп времени, что несколько укорачивает строку вывода;
- **-tt** – выводит неформатированный штамп времени в виде «СЕКУДНЫ.МИКРОСЕКУНДЫ»;
- **-v, -vv** – управляет выводом дополнительной информации (такой, как время жизни пакета TTL, идентификационная информация, и т. д.);
- **-x** – выводит после информации о заголовках каждый пакет в шестнадцатеричном виде (информация заголовка для сокращения записи в примере обрезана):

 # tcpdump -c 3 -x

```
tcpdump:              listening              on              ed1
10:41:47.039140 vipdosug2.masterhost.ru.http > d4.host.ru.4366: ……
                  4500  05dc  aa87  4000  3006  3d7b  d910  1222
                  c3a1  ae45  0050  110e  80b7  dcbd  db29  1b8c
                  5010  e420  2fe5  0000  c789  a882  0c6b  4036
                                    b3ad        5b4e        3520
10:41:47.303925 d4.host.ru.4366 > vipdosug2.masterhost.ru.http: ……
                  4500  0028  5a91  4000  7f06  4425  c3a1  ae45
                  d910  1222  110e  0050  db29  1b8c  80b7  c5ed
                                    5010      2238      e1c9      0000
10:41:47.472827 vipdosug2.masterhost.ru.http > d4.host.ru.4366: ……
                  4500  05dc  b771  4000  3006  3091  d910  1222
                  c3a1  ae45  0050  110e  80b7  e271  db29  1b8c
                  5010  e420  1384  0000  dcfb  53f7  745c  b10b
                10db e090 386a
```

Перечисленные выше параметры позволяют организовать вывод информации в более удобном виде. Следующие опции, задаваемые в параметре «expression», позволяют ограничить выборку пакетов определенными условиями. Рассмотрим только наиболее часто используемые на конкретных примерах.

Ограничить собираемую информацию конкретным адресом позволяет следующая команда:

```
# tcpdump host
```

При этом будут выбираться только пакеты, источником или приемником которых является . Конкретизировать направление передачи можно, указав дополнительный параметр:

```
# tcpdump {src | dst} host
```

Таким же образом просмотр пакетов можно ограничить по тому или иному порту или по конкретному протоколу:

```
# tcpdump [{src | dst}] port
```

```
# tcpdump {tcp | udp | icmp}
```

Ограничить выборку конкретной подсетью позволяет ключевое слово «net». Допускается использование как десятично-точечной нотации, так и нотации CIDR:

```
# tcpdumt net 192.168.0.0 mask 255.255.255.0
```

```
# tcpdumt net 192.168.0.0/24
```

Наибольшую ценность имеет возможность задавать выражения выборки. Запись вида proto[byte] позволяет выбрать конкретный байт в заголовке протокола proto. К полученному байту можно применить операции сравнения (=, !=, <, >), а также бинарные операции «&» (AND), «|» (OR) и другие. Например, чтобы выбрать пакеты, в которых установлены флаги SYN или ACK в заголовке TCP-протокола, можно воспользоваться командой:

```
# tcpdump 'tcp[13] & 18 != 0'
```

Кавычки используются, чтобы не позволить командной оболочке самой интерпретировать специальные символы типа «&». В приведенном выше примере мы выбираем 14-й байт TCP-заголовка, содержащий флаги (не забывайте, что нумерация начинается с нуля), и выполняем операцию логического побитного сложения его содержимого с числом «10010» (десятичное 18), то есть выделяем разряды, соответствующие флагам ACK и SYN. Если хотя бы один из них установлен, то результат не будет равен нулю, и информация по данному пакету будет выведена на экран.

Следующие две команды позволят выбирать пакеты, у которых установлены все флаги (первая) и не установлено ни одного (вторая):

```
# tcpdumt 'tcp[13] & 63 = 63'
```

```
# tcpdump 'tcp[13] & 63 = 0'
```

Еще два полезных параметра: «-w file» и «-r file». Эти ключи позволяют соответственно писать информацию в файл file и читать ранее сохраненные данные из файла. Эта возможность позволяет, например, включить на ночь запись в файл пакетов со всеми флагами, установленными в единицу, а затем проанализировать полученный файл. Однако нужно помнить, что при интенсивной работе размер такого файла может расти очень быстро и заполнить все пространство на диске.

Помимо перечисленных, tcpdump имеет еще целый ряд возможностей, однако они не имеют прямого отношения к материалу данной статьи, и потому рассматриваться здесь не будут.

Использование ipfw для защиты от сканирования

Вот мы и подошли к основному вопросу статьи. Дальнейшее изложение предполагает, что читатель знаком с вопросами установки и настройки пакетного фильтра ipfw, входящего в состав FreeBSD.

В данной статье остановимся лишь на тех особенностях, которые будут использоваться нами для построения правил защиты. Все примеры относятся к версии ipfw, используемой начиная с 5-й версии FreeBSD. В более ранних версиях некоторые особенности могут не работать или иметь несколько отличный от приведенного в статье синтаксис.

В общем виде правило ipfw задается следующим образом:

```
Номер Действие Протокол from Источник to Приемник [Опции]
```

Интерес для нас представляет секция Опции, которая может содержать один или несколько следующих параметров:

- **via интерфейс** – имя интерфейса системы (например, сетевой карты). Если этот параметр задан, правило будет применяться только к пакетам, проходящим через указанное устройство;
- **{in | out}** – опционально может быть указано направление прохождения пакета (in – входящий по отношению к машине, на которой работает ipfw; out – исходящий; по умолчанию учитываются как входящие, так и исходящие пакеты);
- **frag** – правилу будут удовлетворять только фрагментированные пакеты (кроме первого фрагмента);
- **icmptypes types** – тип icmp-пакетов (types может иметь следующие значения: 0 – echo replay, 3 – destination unreach, 5 – redirect, 8 – echo request, 11 – time-to-live exceeded, 12 – IP-header bad, 15 – information request, 16 – information replay и другие);
- **iplen length** – длина IP-пакета;
- **established** – TCP-пакеты с установленными флагами RST или ACK, то есть пакеты, принадлежащие установленному соединению;
- **ipttl ttl** – пакеты с определенным временем жизни (time-to-live, ttl);
- **setup** – пакеты, запрашивающие установление соединения (с установленным битом SYN, но без флага ACK);
- **limit {src-addr | src-port | dst-addr | dst-port} N** – вводит ограничение на число одновременных соединений, удовлетворяющих правилу. Все последующие (свыше N) соединения будут считаться не соответствующими данному правилу. Признаки src-addr, src-port, dst-addr и dst-port указывают, что учитываются пакеты с одного IP-адреса источника, с порта источника, на один адрес приемника или на порт приемника соответственно.

В качестве примера приведу правило, позволяющее ограничить число TCP-соединений на данную машину с одного IP-адреса (признак src-addr):

```
# ipfw add number allow tcp from any to me setup limit src-addr 5
```

Данное правило пропустит только первые 5 запросов на соединение (признак setup), поступившие с одного IP-адреса, а для остальных пакетов с этого адреса проверка на соответствие правилам будет продолжена, и в случае закрытого брандмауэра они будут отброшены после проверки всей цепочки правил.

- **uid user** – пакеты, посланные указанным пользователем или адресованные указанному пользователю;

- **gid group** – пакеты, посланные пользователем, принадлежащим группе group или адресованные пользователю этой группы;
- **tcpwin win** – TCP-пакеты с указанным размером окна приема (window);
- **tcpflags флаги** – задает перечень флагов TCP-пакета, которые должны быть установлены. Допускаются значения fin, syn, rst, psh, ack и urg. Символ «!» перед именем флага означает, что данный флаг должен быть сброшен.

Любой из этих параметров может предваряться служебным словом «not», которое инвертирует значение параметра.

Как мы видели, один из способов сканирования портов удаленной машины – посылка пакетов с установленным флагом FIN. Такое сканирование может быть выполнено, например, с помощью команды:

```
# nmap –sF
```

В нормальной ситуации данный флаг сигнализирует об окончании сеанса связи, но поскольку в данном случае связь не установлена, то запрошенная система отвечает сообщением об ошибке, по наличию которого хакер и определяет, открыт сканируемый порт или нет. Защититься от подобного сканирования можно с помощью следующего правила:

```
# ipfw add number reject tcp from any to any not established tcpflags fin
```

В соответствии с ним все TCP-пакеты с установленным флагом FIN, но не принадлежащие установленным соединениям (not established) будут отбрасываться.

Сканирование с установкой или сбросом всех флагов (параметры nmap –sX и –sN соответственно) можно сделать бесполезным, добавив в цепочку следующие два правила:

```
# ipfw add number reject tcp from any to any tcpflags fin, syn, rst, psh, ack, urg

# ipfw add number reject tcp from any to any tcpflags !fin, !syn, !rst, !psh, !ack, !urg
```

То есть если в пакете все флаги установлены или, наоборот, сброшены, то пакет будет отброшен соответствующим правилом.

От сканирования, выполняемого с параметрами nmap -sT и -sS («соединение» и «полусоединение»), нельзя закрыться запрещающим правилом, подобно изложенному выше, поскольку при данных методах сканирования соединение устанавливается (или начинает устанавливаться) в соответствии с протоколом TCP. Естественно, мы не можем запретить все пакеты, запрашивающие соединение с тем или иным портом. Однако поскольку и реакция на такие пакеты будет стандартной, то злоумышленник уже не сможет получить дополнительную информацию о системе, такую, например, как версия ОС. Кроме того, интенсивность сканирования можно существенно снизить, установив ограничение «limit» на число соединений с одного адреса (см. выше).

Кроме того, нельзя забывать, что грамотная политика фильтрации пакетов, когда разрешаются только соединения с нужными портами и с ограниченного числа адресов, позволит сделать сканирование практически бесполезным.

Раз уж мы взялись защищать нашу сеть от хакеров, заодно добавим правило против спуфинга (подмены IP-адресов на легальные). Смысл спуфинга заключается в том, что злоумышленник отсылает пакеты от имени доверенного хоста, подменяя IP-адрес отправителя. Например, мы можем закрыть доступ к порту 110 (POP3) со всех адресов, кроме локальной подсети 193.163.0.0/24. Но если хакер пошлет на порт 110 пакет от имени пользователя локальной сети, например, подставив адрес 193.163.0.12, то он получит доступ к порту. Конечно, исключить такую возможность можно грамотной маршрутизацией, но подстраховаться никогда не вредно. Добавим в цепочку еще одно правило:

```
# ipfw add 10007 deny ip from any to any not verrevpath in
```

В результате пакеты, пришедшие с интерфейса, отличного от того, куда они были бы направлены маршрутизатором, будут отброшены. Это исключит возможность выдать себя за легального пользователя локальной сети, подключившись извне (конечно, только в том случае, когда локальная сеть и «внешний мир» подключены через разные интерфейсы).

Кроме того, никогда не лишне (особенно когда речь идет о безопасности) вести журнал всех «злонамеренных» пакетов. Для этого правило нужно снабдить ключевым словом «log» после действия, которое должно быть выполнено по отношению к пакету, удовлетворившему данному правилу, например:

```
# ipfw add 10007 deny log ip from any to any not verrevpath in
```

Теперь, если пакет будет получен с «неправильного» интерфейса, то syslogd получит сообщение уровня LOG_SECURITY, которое будет обработано в соответствии с настройками в /etc/syslog.conf.

Кроме того, ядро системы должно быть собрано с опцией «IPFIREWALL_VERBOSE». Чтобы снизить вероятность переполнения диска журнальной информацией, предусмотрен механизм ограничения числа одинаковых записей. Ограничение по умолчанию задается опцией ядра, например:

```
option     "IPFIREWALL_VERBOSE_LIMIT=100"
```

В этом случае в журнал будут записаны только 100 записей, соответствующих правилу. Кроме того, это значение можно переопределить для конкретного правила, задав секцию logamount number после ключевого слова log:

```
# ipfw add number reject log logamount 5 tcp from any to any not established tcpflags fin
```

Журналирование – функция очень полезная, однако если у вас недостаточно места на диске, то использовать ее следует очень осторожно, поскольку одним из распространенных последствий DoS-атак является переполнение раздела файловой системы журнальной информацией и, как следствие, – отсутствие какого бы то ни было контроля за попытками подключения к системе и отказ таких служб, как сервер электронной почты, которые требуют дискового пространства для своей работы, а порой и полная неработоспособность системы. К слову заметим, что последствия этого можно снизить правильной разбивкой диска на разделы, когда не особо важная, но быстрорастущая информация (от ipfw, apache, squid и т. д.) записывается на свой раздел и не влияет на разделы, хранящие, например, логи авторизации и спул электронной почты.

Итак, мы создали набор базовых правил, которые существенно усложнят сканирование нашей системы. Теперь нам нужно записать эти правила в конфигурационный файл, чтобы при перезагрузке системы они автоматически активировались. Для ipfw есть две возможности увековечить наши правила.

Во-первых, это конфигурационный файл /etc/rc.firewall. По умолчанию в нем уже содержится два правила с номерами 100 и 200 (иногда и правило 300), которые служат для обеспечения доступа к «внутреннему» интерфейсу системы lo0 и ограничивают доступ к «внутренним» адресам системы 127.0.0.0/8. По аналогии с ними можно дописать в этот файл и наши правила в виде:

```
$fwcmd add ПРАВИЛО
```

Например, упомянутые выше два правила заданы следующим образом:

```
$fwcmd add 100 pass all from any to any via lo0

$fwcmd add 200 deny all from any to 127.0.0.0/8
```

Однако нужно иметь в виду, что /etc/rc.firewall – системный файл, и его лучше не изменять (это позволит избежать проблем, например, при обновлении системы с помощью cvsup).

Более правильным является конфигурация ipfw как пользовательского. Для этого в файле /etc/rc.conf поменяем тип брандмауэра:

```
firewall_type="/etc/ipfw.conf"
```

Теперь создадим файл ipfw.conf в папке /etc (впрочем, имя файла может быть любым удобным для вас) и поместим в него наши правила так, как если бы вводили их с консоли, только без имени программы ipfw. То есть выглядеть этот файл будет примерно так:

```
# Запрет X-сканирования:
```

```
add 1001 reject log tcp from any to any tcpflags fin, syn, rst, psh, ack, urg

# Запрет N-сканирования:

add 1002 reject log tcp from any to any tcpflags !fin, !syn, !rst, !psh, !ack, !urg

# Запрет FIN-сканирования:

add 1003 reject log tcp from any to any not established tcpflags fin

# Защита от спуфинга

add 1004 deny log ip from any to any not verrevpath in

# Ограничение числа одновременных соединений:

add 1005 allow ip  from any to any  setup limit src-addr 10
```

Комментарии, как обычно, предваряются символом «#». Теперь при загрузке системы автоматически будут включаться правила сначала из rc.firewall, затем из пользовательского файла (в нашем случае ipfw.conf).

На первый взгляд запрет нестандартных пакетов может показаться излишним, поскольку в случае «закрытого» брандмауэра они в любом случае будут отброшены последним запрещающим правилом. Однако их явный запрет позволит вести запись попыток сканирования тем или иным методом, что даст возможность вовремя обнаружить чей-то интерес к вашей системе и предпринять ряд превентивных мер. Кроме того, размещение этих правил в начале цепочки позволит несколько разгрузить систему в случае массированного сканирования, поскольку нестандартные пакеты не станут проверяться на соответствие всем правилам цепочки, а будут отброшены в ее начале.

Итак, мы рассмотрели основные пути защиты сети, первая линия обороны которой построена на базе FreeBSD с брандмауэром ipfw. Безусловно, соответствующей настройки еще не достаточно, чтобы спать спокойно. Ежедневный анализ log-файлов, непрерывное повышение общей грамотности в области протоколов, межсетевых экранов и т. д. – вот базовые составляющие безопасности. Надеюсь, что эта статья позволит вам более осознанно строить защиту вашей сети.

Дополнительные материалы:

1. Чубин И. Ipfw и управление трафиком в FreeBSD. – Журнал «Системный администратор», №6, июнь 2003 г. – 26-34 с.
2. Страницы справочного руководства man ipfw, man tcpdump, man nmap.