Скрытые возможности IPTables

С помощью этого мощного набора расширений для iptables вы сможете строить свои правила основываясь на анализе содержимого пакетов, диапазона портов и даже создавать ловушки для злоумышленников.

Iptables в Linux позволяет строить весьма мощные брандмауэры, ничуть не уступающие по своим характеристикам многим коммерческим системам защиты. По сути своей, iptables основывается на фильтрации пакетов, проходящих через соединение, и в соответствии с набором правил определяет ту или иную реакцию брадмауэра на эти пакеты. В самых простых случаях iptables может использоваться для того, чтобы сбросить одни пакеты и пропустить другие. При этом обычно анализируются IP-адрес пакета, номер порта и направление движения пакета. Кроме того iptables может анализировать статус пакета (NEW, ESTABLISHED, RELATED и пр. прим. перев.)

Простая и очень эффективная возможность блокировки входящего трафика, инициированного извне, совместно с функциями трансляции сетевых адресов (NAT) дает возможность пользователям свободного выхода во "внешний мир" и надежно защищает их от вторжений извне. Обычно такие наборы правил несколько упрощены, и возможно их следует расширить дополнительными фильтрами, но сама концепция остается неизменной.

Iptables может предложить много больше, чем эти простые критерии. Некоторые из дополнений известны довольно широко и даже включаются в состав некоторых дистрибутивов Linux. Другие не так известны, но тем не менее так же заслуживают пристального внимания. Об этих дополнительных возможностях я и собираюсь рассказать вам в этой статье. Для того, чтобы описать все дополнения потребовалась бы целая книга, я лишь хочу побудить вас к дальнейшему самостоятельному их изучению.

Введение в POM

Netfilter состоит из двух больших групп компонентов, первая -- компоненты включенные в состав ядра (и выполняющиеся в пространстве ядра), вторая -- это команды и утилиты

выполняющиеся в пространстве пользователя, сюда можно отнести собственно команду iptables, ряд вспомогательных утилит, библиотеки, страницы справочного руководства и скрипты. К компонентам ядра можно отнести "заплаты" на исходные тексты ядра и дополнительные модули.

Наложение "заплат" на ядро -- задача весьма нетривиальная и для непосвященного может оказаться весьма сложной и запутанной. Ошибки при наложении "заплат" или наложение "заплат" несовместимых с данной версией ядра, могут привести к тому, что ядро невозможно будет собрать или хуже того -- ядро откажется загружаться. Команда разработчиков Netfilter постаралась упростить процесс наложения "заплат", поставляя в составе POM, или Patch-o-matic, сценарии, выполняющие эту работу за нас. POM -- это набор "заплат" на исходные тексты ядра и комплект сценариев, выполняющих наложение этих "заплат", что делает возможным работу с POM даже для новичков.

"Заплаты", входящие в состав POM подразделяются на ряд категорий, в зависимости от хронологии и уровня готовности. Некоторые из них обязательны при каждой установке iptables/Netfilter. Другие -- необязательные или находящиеся на стадии тестирования, но предоставляющие весьма интересные дополнительные возможности, это и есть те самые скрытые возможности, которым посвящена данная статья. В документации к POM они описаны как: "Maybe broken, Maybe cool extensions." ("Возможно неустойчивы в работе. Возможно чертовски хороши." да простит меня читатель за эмоциональный перевод прим. перев.)

Установка POM очень проста: скачайте тарболл с последней версией Patch-o-matic из каталога /pub/patch-o-matic на ftp.netfilter.org, распакуйте и запустите соответствующую команду, само собой разумеется из под пользователя root. Не забудьте указать корректный путь к исходным текстам ядра в параметре KERNEL_DIR:
KERNEL_DIR=/usr/src/linux-2.4 ./runme extra

Сам процесс установки -- интерактивный и достаточно понятный.

Расширение string

Расширение string -- на мой взгляд одно из самых востребованных, среди прочих дополнений из POM. Оно позволяет выполнять фильтрацию пакетов, основываясь на анализе содержимого области данных пакета. Этот модуль имеет очень широкую область применений, но тут главное не перестараться. Как один из вариантов применения, можно привести возможность блокировки скачивания исполняемых файлов в формате ELF через WEB. Известно, что исполняемые файлы в формате ELF начинаются символом с кодом 7Fh, за которым следуют буквы ELF. Таким образом можно проанализировать все пакеты, поступающие из Интернет с исходящего порта 80, на предмет наличия в области данных пакета требуемой комбинации символов. Шестнадцатиричные коды символов должны заключаться между символами "|", например так: |7F|ELF. Если предположить, что в Интернет "смотрит" интерфейс eth0,

то фильтрующее правило может выглядеть примерно так:
iptables -A FORWARD -i eth0 -p tcp --sport 80
-m string --string '|7F|ELF' -j DROP
Синтаксис записи шестнадцатиричных кодов символов, в данном виде, поддерживается iptables, начиная с версии 1.2.8. Если вы используете более раннюю версию, то вам придется прибегнуть к "обману", например:
--string "`dd if=/bin/ls bs=4 count=1 2>/dev/null`"
здесь в критерий попадут первые четыре символа из файла /bin/ls, который является исполняемым файлом формата ELF и который содержит требуемую нам комбинацию символов.

А теперь несколько расширим этот пример. Допустим, что мы абсолютно доверяем содержимому, поступающему с адреса 192.168.0.5 и потому не хотим фильтровать трафик, поступающий с этого адреса. Делается это простым добавлением инвертированного критерия проверки исходящего IP-адреса, например так:
iptables -A FORWARD -i eth0 -p tcp
! -s 192.168.0.5 --sport 80 -m string
--string '|7F|ELF' -j DROP
Однако данные правила не свободны от ошибок. Первое -- приведенный здесь критерий будет срабатывать для любого пакета, в области данных которого содержится указанная комбинация символов, а не только в начале передаваемого исполняемого файла. Это может вызвать ложные срабатывания критерия и привести к блокировке вполне легитимных пакетов. Второе -- если искомая последовательность символов окажется разнесенной по двум, или более, смежным пакетам (начало искомой строки в конце одного пакета, а конец строки в начале другого), то критерий не сработает. Искомая строка должна полностью входить в состав одного пакета.

Таким образом, критерий string может сослужить неплохую службу, но вам не следует забывать об упомянутых выше особенностях. Кроме того, он чувствителен к регистру символов и не сможет среагировать на строку, разнесенную по смежным пакетам.

Расширение mport

Расширение mport позволяет указывать в одном правиле список портов и диапазон портов. Без этого расширения iptables позволяет указывать что-то одно -- либо один порт, либо диапазон портов (хочется упомянуть о существовании расширения multiport, которое уже включено в ядро, однако это расширение позволяет подставить только список портов т.е. не позволяет объединять список портов и диапазон в один критерий прим. перев.). С помощью расширения mport допускается более сложный синтакис. Например: мы хотим разрешить соединения с X-терминалов, Web и почту. Для этого мы теперь можем построить единственное правило, например так:
iptables -A INPUT -p tcp -m mport
--dports 80,110,21,6000:6003 -j ACCEPT
Без расширения mport нам пришлось бы создать несколько правил:

iptables -A INPUT -p tcp --dports 80 -j ACCEPT
iptables -A INPUT -p tcp --dports 110 -j ACCEPT
iptables -A INPUT -p tcp --dports 21 -j ACCEPT
iptables -A INPUT -p tcp --dports 6000:6003 -j ACCEPT
От переводчика:

При использовании расширения multiport нам потребовалось бы написать два правила:
iptables -A INPUT -p tcp -m multiport --dports 21,80,110 -j ACCEPT
iptables -A INPUT -p tcp --dports 6000:6003 -j ACCEPT

Создание единственного правила вместо четырех безусловно влечет за собой повышение пропускной способности брандмауэра и снижение нагрузки на систему, поскольку теперь пакеты обрабатываются одним правилом вместо четырех. Кроме того, набор правил можно упростить, благодаря тому, что теперь сервисы, требующие идентичной обработки легко группируются в одно правило.

Расширение time

Расширение time позволяет строить логику критерия, основанную на текущем времени суток и дне недели. Например, можно ограничить доступ к WEB-серверу в определенное время суток или перенаправить трафик на зеркалирующий WEB-сервер в часы проведения плановых профилактических работ на основном сервере. Следующий пример иллюстрирует ограничение доступа к WEB-серверу по пятницам, с 4 до 6:30 часов утра, на время проведения профилактических работ:
iptables -A INPUT -p tcp -d 80 -m time
--timestart 04:00 --timestop 06:30 --days Fri
--syn -j REJECT
Следует отметить, что все три ключа -timestart, -timestop и -days обязательно должны быть включены в правило. Таким образом, если необходимо построить аналогичное правило, которое не зависит от дня недели, то вам придется явно указать все семь дней недели.

Создание ловушек

Едва ли кто нибудь из нас желал бы угодить в ловушку, если конечно вы цените свою жизнь. Расширение TARPIT представляет собой эквивалент ловушки -- попавшему в нее не удастся быстро выбраться на свободу. Если вы были настолько неблагоразумны, что попытались установить соединение с портом-ловушкой, то обнаружите, что закрыть такое соединение (и освободить тем самым системные ресурсы) не так-то просто.

Чтобы добиться такого эффекта, iptables подтверждает запрос на TCP/IP соединение и устанавливает размер окна равным нулю, что вынуждает атакующую систему

прекратить передачу данных -- очень напоминает нажатие комбинации Ctrl+S в терминале. Любые попытки атакующего закрыть соединение игнорируются, таким образом соединение остается открытым, пока не истечет срок тайм аута (обычно 12-24 минуты), что в свою очередь приводит к расходу системных ресурсов атакующей системы (но не системы-ловушки). Правило, создающее ловушку может выглядеть примерно так:
iptables -A INPUT -p tcp -m tcp -dport 80 -j TARPIT
Едва ли стоит использовать conntrack и TARPIT на одной и той же системе, особенно если ожидается большое число соединений, попавших в ловушку. Каждое такое соединение будет расходовать ресурсы conntrack.

Еще один пример -- как можно оконфузить злоумышленника, имитируя поведение Microsoft Windows. В ответ на попытку сканирования портов netbios система может отвечать системе атакующего, а затем переводить эти соединения на TARPIT. Атакующий будет тратить время впустую, полагая, что порты открыты и пытаясь установить соединение. Он будет крепко раздосадован долгим ожиданием ответа и явно безумным поведением атакуемой системы. Правило, которое дает такой эффект может выглядеть следующим образом:
iptables -A INPUT -p tcp -m tcp -m mport
 --dports 135,139,1025 -j TARPIT
Еще один пример использования TARPIT -- установить ловушки на ВСЕ порты, кроме определенных вами. Это опять-таки будет вводить в заблуждение посторонних, демонстрируя им, что все порты открыты и заставляя их тратить свое время на попытки установить соеднение. Более того, это предотвращает возможность определения типа операционной системы на системе-ловушке с помощью tcpdump. В данном примере легитимными считаются только сервисы WEB и E-MAIL, любые другие соединения будут "срываться" в ловушку.
iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --dport 25 -j ACCEPT
iptables -A INPUT -p tcp -m tcp -j TARPIT
На www.spinics.net/lists/netfilter/msg17583.html вы найдете интересный пример из реальной жизни, когда расширение string и TARPIT сослужили неплохую службу системному администратору (не мне).

Срабатывание критерия с заданной вероятностью

Расширение random позволяет строить критерии, срабатывание которых зависит от заданной вероятности. Вы можете построить такой критерий, который будет срабатывать с вероятностью в диапазоне от 0% до 100% случаев. Это расширение можно использовать, например, для эмуляции неустойчивости соединения с сервером или для равномерного распределения нагрузки по нескольким зеркалам. Пример, приведенный ниже, демонстрирует распределение WEB-трафика по трем серверам. Первое правило отправляет 33% соединений на адрес 192.168.0.100. Следующее правило отправляет 33% от общего числа соединений (50% от оставшихся 66% после

первого правила прим. перев.) на адрес 192.168.0.101 и последнее правило отправляет все остальные соединения на адрес 192.168.0.102:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp
 --dport 80 --syn -m random --average 33
 -j DNAT --to-destination 192.168.0.100:80

iptables -t nat -A PREROUTING -i eth0 -p tcp
 --dport 80 --syn -m random --average 50
 -j DNAT --to-destination 192.168.0.101:80

iptables -t nat -A PREROUTING -i eth0 -p tcp
 --dport 80 --syn -j DNAT
 --to-destination 192.168.0.102:80
```

Как и прежде предполагается, что в Интернет "смотрит" eth0.

И многое, многое другое

POM предоставляет огромное количество расширений. Я описал здесь лишь очень незначительную их часть. Просто запустите сценарий runme и просмотрите описания к "заплаткам", по мере их появления на экране. Ниже приводится кое что еще из того, что вы сможете обнаружить:

Трассировка соединений для RSH, MMS (media streaming), PPTP, Quake, RPC и Talk.
Расширенная поддержка доступа к конфигурации и к информации о состоянии через файловую систему /proc.
Расширенная поддержка особенностей IPv6.
Манипуляции с полем TTL и другими полями в IP-пакетах.
Более тонкое управление соединениями через NAT.
Ограничение трафика установкой квот и пропускной способности канала.
Блокировка попыток определения типа операционной системы (Anti-OS fingerprinting) и обнаружение сканирования портов. Маркировка соединений (и проверка маркировки).

Справочные материалы

Наложение "заплат" из POM не означает установку описаний новых возможностей в страницы справочного руководства по iptables, поэтому вам придется обращаться непосредственно к сопровождающей документации. Базовый синтаксис того или иного расширения вы сможете получить с помощью встроенной подсказки iptables. Например, iptables -m random --help выведет обычное, для iptables, справочное сообщение, но с дополнительной справкой по расширению random в конце. Аналогичную справку вы получите и для других расширений.

Вы так же можете обращаться к файлам справки по конкретным модулям, которые вы найдете в дереве каталогов Patch-o-matic. Например, справку по модулю random вы

найдете в base/random/help. Аналогичные файлы-справки существуют и для других "заплат".

И, наконец, можете обратиться на сайт Netfilter, www.netfilter.org/patch-o-matic, где вы всегда найдете описание для каждой из "заплат", включенной в состав POM.

Установка новых модулей iptables

По большей части, расширения для iptables состоят из двух частей -- "заплаты" на ядро и вспомогательной библиотеки, которая используется командой iptables в пространстве пользователя. Детальное описание процедуры добавления POM модулей вы найдете на www.lowth.com/howto/add-iptables-modules.php. В двух словах процесс установки выглядит так: вам необходимо обновить систему, загрузить последний Patch-o-matic, наложить "заплаты" на ядро (с помощью сценария runme), пересобрать и установить пропатченное ядро и пересобрать и установить iptables.

Заключение

Вы уже наверняка убедились, что Netfilter представляет собой замечательный инструментальный набор, пригодный для построения эффективных брандмауэров, но в большинство дистрибутивов Linux попадают далеко не все расширения. Patch-o-matic, с возможностью практически автоматического процесса наложения "заплат" на ядро, позволяет администраторам расширять функциональность своих брандмауэров, .

Заканчивая свою статью хочется еще раз обратить ваше внимание: мы с вами увидели, что iptables/Netfilter предлагает множество интересных дополнений, которые не видны на первый взгляд. Что справедливо для любого другого программного обеспечения. Это одна из замечательных особенностей open-source -- никто ни от кого ничего не прячет, просто нужно поискать как следует!

Благодарности

Спасибо Jane Lowth за красивые рисунки пингвина Тукса (Tux).Расширение time

оригинал: http://ftp.linux.kiev.ua/pub/docs/mirrors/gazette.linux.ru.net/rus/articles/iptables-treasures.html