

Linux IPCHAINS-HOWTO

Rusty Russell

v1.0.8, Tue Jul 4 14:20:53 EST 2000

Перевод на русский язык: Н. Малых nmalykh@bilim.com

В этом документе рассказано, как получить, установить и настроить цепочки брандмауэра IP в среде Linux, а также рассмотрены некоторые идеи, связанные с использованием цепочек.

Оглавление

Введение	2	Использование ipchains-restore	15
Что такое ipchains?	2	Разное	16
Зачем использовать ipchains?	2	Как организовать правила для брандмауэра	16
Какое ядро следует использовать?	2	Что не следует фильтровать	16
Где найти документ?	2	Пакеты ICMP	16
Основы фильтрации пакетов	2	TCP-соединение с DNS (сервер имен)	16
Что делает фильтр пакетов?	2	Проблемы с FTP	16
Зачем фильтровать пакеты?	2	Фильтрация Ping of Death	17
Как организовать фильтрацию?	3	Фильтрация Teardrop и Bonk	17
Ядро с фильтрацией пакетов	3	Фильтрация Fragment Bomb	17
ipchains	3	Изменение правил брандмауэра	17
Установка постоянных правил	3	Как организовать защиту от обманок IP Spoof?	17
Маршрутизация, маскардинг, пересылка	4	Дополнительные возможности	18
Как организовать маскардинг (Masquerading)	4	SPF - Stateful Packet Filtering	18
Средства настройки цепочек	4	ftp-data hack Майкла Азенстайна	18
Установка параметров общего назначения для брандмауэра	4	Возможности будущих версий	18
Частная сеть – традиционный посредник (Proxy)	4	Проблемы общего плана	18
Частная сеть – прозрачный посредник	5	Команда ipchains -L приводит к «умиранию» системы!	18
Частная сеть – маскардинг	5	Инверсия не работает!	18
Сеть общего пользования	6	Маскардинг/пересылка не работает!	18
Ограниченный доступ к внутренним службам	6	-j REDIR не работает!	19
Дополнительные сведения о маскардинге	6	Не работают шаблоны интерфейсов!	19
Цепочки IP Firewalling	6	TOS не работает!	19
Как пакеты проходят через фильтры	6	Не работают ipautofw и ipportfw!	19
Использование ipchains	7	xosview не поддерживается!	19
Что будет происходить при загрузке операционной системы	7	Ошибка Segmentation Fault при использовании -j REDIRECT!	19
Операции с одним правилом	8	Я не могу установить тайм-аут для маскардинга!	19
Спецификации фильтров	8	Я хочу использовать брандмауэр для протокола IPX!	19
Задание IP-адресов отправителя и получателя	8	Реальный пример	19
Обращение правила	8	Схема сети	19
Указание протокола	8	Задачи	20
Задание портов UDP и TCP	9	Прежде, чем начать фильтрацию пакетов	20
Задание типа и кода ICMP	9	Фильтрация для транзитных пакетов	20
Задание интерфейса	9	Установка переходов из цепочки forward	20
Обработка пакетов TCP SYN	9	Определение цепочки icmp-асс	21
Обработка фрагментов	9	Доступ изнутри к DMZ (серверы)	21
Результаты работы фильтров	10	Доступ извне к DMZ	21
Задание действия (Target) для фильтра	10	Доступ изнутри во внешние сети (Bad)	21
Протоколирование (Logging) пакетов	11	Доступ из DMZ во внутреннюю сеть	21
Управление типом обслуживания (TOS)	12	Доступ из DMZ во внешнюю сеть	22
Маркировка пакетов	12	Доступ извне во внутреннюю сеть	22
Операции с целой цепочкой	12	Фильтрация пакетов для брандмауэра	22
Создание новой цепочки	12	Внешний интерфейс (Bad)	22
Удаление цепочки	12	Интерфейс DMZ	22
Уничтожение (Flushing) цепочки	13	Внутренний интерфейс (Good)	22
Просмотр списка цепочек	13	Заключение	23
Сброс (обнуление) счетчиков	13	Различия между ipchains и ipfwadm	23
Установка политики для цепочек	13	Краткая справочная таблица	23
Операции с маскардингом	14	Примеры трансляции команд ipfwadm	24
Проверка пакета	14	Использование сценария ipfwadm-wrapper	24
Работа с несколькими правилами сразу и определение результата	14	Приложение. Благодарности	24
Полезные примеры	14	Переводы	24
Использование ipchains-save	15		

Введение

Этот документ является переводом Linux IPCHAINS-HOWTO с учетом опыта реального использования ipchains. Оригинальный документ вы сможете на указанных ниже сайтах (см. параграф Где найти документ?), свежий вариант перевода – на сайте <http://www.bilim.com>. Для понимания данного документа полезно будет также прочесть Linux NET-3-HOWTO, IP-Masquerading HOWTO, PPP-HOWTO, Ethernet-HOWTO и Firewall HOWTO. (эти документы можно найти в alt.fan.bigfoot FAQ).

Если вы хорошо знакомы с фильтрацией пакетов, после прочтения параграфа Зачем использовать ipchains? Можно сразу перейти к разделу Цепочки IP Firewalling.

Если вы раньше использовали ipfwadm, прочтите раздел Введение (параграф Какое ядро следует использовать?) и перейдите к приложениям Различия между ipchains и ipfwadm и Использование сценария ipfwadm-wrappers.

Что такое ipchains?

Linux ipchains представляет собой брандмауэр Linux IPv4 (в основном «списанный» с аналогичного кода BSD) и используемый взамен ipfwadm (перепев BSD). Для использования ipchains требуется ядро версии 2.1.102 или выше.

Зачем использовать ipchains?

Старый код брандмауэра Linux не может работать с фрагментами, использует 32-битовые счетчики (по

Основы фильтрации пакетов

Что делает фильтр пакетов?

Весь трафик в сети передается в форме пакетов. Например, загрузка документа размером 50К, может обеспечиваться в форме передачи 36 пакетов размером по 1460 байтов.

В начале каждого пакета содержится служебная информация (заголовок), описывающая тип пакета, адреса отправителя и получателя, а также другие детали. Остальная часть пакета содержит реальные данные, которые требуется передать из одной точки сети в другую (часто эту часть называют телом пакета).

Некоторые протоколы (такие, как TCP), используемые для передачи web-трафика, электронной почты и удаленного входа в систему, работают на основе концепции соединений – прежде, чем начнется передача реальной информации, происходит обмен служебными пакетами (они используют специальные заголовки) ожидая организации соединения. Обмен данными начинается лишь после организации соединения.

Фильтр пакетов представляет собой программный код, обеспечивающий просмотр заголовков пакетов на предмет принятия решения о передаче или отбрасывании каждого пакета. Фильтр обеспечивает принятие решения об игнорировании пакета (как будто его и не было), принятии его (пакет проходит через фильтр) или отказе от его приема (подобно игнорированию, но отправитель пакета получает информацию об отказе).

В среде Linux фильтрация пакетов встроена в ядро операционной системы и существуют возможности «тонкой обработки» пакетов, но основной задачей

крайней мере на платформе Intel), не позволяет указать протоколы, отличные от TCP, UDP и ICMP, не может делать больших изменений по частям, не позволяет задавать инверсные правила, имеет целый ряд ошибок и достаточно сложен в управлении (это часто приводит к ошибкам).

Какое ядро следует использовать?

Код ipchains поддерживается ядрами, с номером версии 2.1.102 и выше. Для ядер версии 2.0 существуют специальные «заплаты», которые можно найти в сети Internet. Если ваше ядро имеет номер версии выше, нежели номер версии заплата, это не страшно (например, заплата версии 2.0.34 может использоваться с ядром 2.0.35). Поскольку заплата версии 2.0 несовместима с заплатами ipportfw и ipautofw, не рекомендуется применять этот «патч».

Где найти документ?

Официальный текст оригинального документа можно найти на серверах Penguin Computing (<http://netfilter.filewatcher.org/ipchains>), SAMBA Team (<http://www.samba.org/netfilter/ipchains>) и Jim Pick (<http://netfilter.kernelnotes.org/ipchains>).

Для обсуждения, рассылки сообщений об ошибках и новых разработках поддерживается список рассылки. Подписаться на рассылку можно на сервере east.balius.com.

фильтра пакетов по-прежнему остается принятие решения и принятии или отбрасывании пакетов.

Зачем фильтровать пакеты?

Контроль. Безопасность. Бдительность.

Контроль:

При использовании Linux-системы для соединения внутренней сети с другой сетью (например, Internet) вы можете разрешить прохождение некоторых типов трафика и запретить трафик других типов. Например, вы можете предотвратить передачу пакетов по каким-либо адресам внешней сети. Можно также с помощью фильтров отключить загрузку рекламных баннеров на Web-страницах, установив фильтр по адресам отправителей в баннерных сетях типа doubleclick.net (существуют и другие способы избавиться от ненужной рекламы).

Безопасность:

Когда Linux-машина является единственной дверью между хаосом Internet и вашей упорядоченной локальной сетью, хорошо бы знать, кто пытается взломать эту дверь. Например, вы можете разрешить беспрепятственный проход во внешнюю сеть, но вряд ли у вас появится желание пропускать в сеть пакеты хорошо известной атаки Ping of Death (смертельный удар) от злоумышленников во внешней сети. Вы можете не разрешить доступ извне к вашей Linux-машине по протоколу telnet даже при использовании паролей для всех пользователей. В этом случае все

пакеты организации соединений Telnet будут отвергаться на входе.

Бдительность:

В некоторых случаях плохо настроенная машина в локальной сети может слать ненужные пакеты во внешнюю сеть. Очень эффективно в таких случаях использовать фильтр пакетов для уведомления администратора о подобном поведении локальной машины.

Как организовать фильтрацию?

Ядро с фильтрацией пакетов

Для использования ipchains вам потребуется ядро с поддержкой цепочек брандмауэра IP. Для того, чтобы узнать, поддерживает ли ваше ядро эти цепочки, проще всего проверить наличие файла /proc/net/ip_fwchains. Если этот файл существует, цепочки поддерживаются вашим ядром.

Если текущее ядро не поддерживает цепочек, нужно установить новое ядро, которое может работать с цепочками. Загрузите исходные тексты нового ядра (номера версий ядра с поддержкой цепочек начинаются с 2.1.102, а для предыдущих версий ядра существуют специальные заплатки - патчи). При использовании заплат настройте конфигурационные параметры ядра в соответствии с приведенными ниже рекомендациями. Если вы не знаете, как задать конфигурационные параметры ядра, обратитесь к документу Kernel-HOWTO.

Для ядер 2.0 нужно установить следующие опции:

```
CONFIG_EXPERIMENTAL=y
CONFIG_FIREWALL=y
CONFIG_IP_FIREWALL=y
```

Различия между ipchains и ipfwadm, а приложение Использование сценария ipfwadm-wrapper содержит более детальное описание ipfwadm.

Установка постоянных правил

Текущие параметры брандмауэра сохраняются в ядре и при перезагрузке операционной системы теряются. Для организации постоянных правил рекомендуется

```
#!/bin/sh
# Script to control packet filtering.
# If no rules, do nothing.
[ -f /etc/ipchains.rules ] || exit 0
case "$1" in
    start)
        echo -n "Turning on packet filtering:"
        /sbin/ipchains-restore < /etc/ipchains.rules || exit 1
        echo 1 > /proc/sys/net/ipv4/ip_forward
        echo "."
        ;;
    stop)
        echo -n "Turning off packet filtering:"
        echo 0 > /proc/sys/net/ipv4/ip_forward
        /sbin/ipchains -F
        /sbin/ipchains -X
        /sbin/ipchains -P input ACCEPT
        /sbin/ipchains -P output ACCEPT
        /sbin/ipchains -P forward ACCEPT
        echo "."
        ;;
    *)
        echo "Usage: /etc/init.d/packetfilter {start|stop}"
        exit 1
        ;;
esac
exit 0
```

```
CONFIG_IP_FIREWALL_CHAINS=y
```

Для ядер 2.1 и 2.2 нужно установить опции:

```
CONFIG_FIREWALL=y
CONFIG_IP_FIREWALL=y
```

Цепочки ipchains обмениваются информацией с ядром и говорят ему, какие пакеты следует фильтровать.

ipchains

ipchains обеспечивает механизмы вставки и удаления правил в раздел фильтрации пакетов ядра операционной системы. Заданные цепочки сохраняются только в течение сеанса работы операционной системы (до следующей перезагрузки компьютера). Операции по заданию постоянных правил фильтрации приведены в следующем параграфе. Постоянные правила сохраняют свою актуальность и после перезагрузки операционной системы на компьютере.

ipchains используется взамен механизма ipfwadm, который поддерживался для старых вариантов IP Firewall. На FTP-сервере ipchains вы сможете найти множество полезных сценариев для работы с цепочками фильтрации пакетов:

<http://netfilter.filewatcher.org/ipchains/ipchains-scripts-1.1.2.tar.gz>

В приведенном файле содержится shell-сценарий ipfwadm-wrapper, позволяющий организовать фильтрацию пакетов привычным способом. Обычно этот сценарий используется для быстрого обновления систем, на которых использовался механизм ipfwadm. Если вы решили пойти этим путем, остальную часть данного документа можно не читать (отметим, что этот вариант является более медленным и не обеспечивает проверки аргументов). Основные различия между механизмами описаны в приложении

использовать сценарии ipchains-save и ipchains-restore. Для того, чтобы установить постоянные правила, сохраните требуемый набор правил от имени пользователя root с помощью команды:

```
# ipchains-save > /etc/ipchains.rules
#
```

После этого создайте сценарий, подобный приведенному ниже:

Проверьте работу сценария и обеспечьте его автоматическое выполнение при загрузке ОС. Одним из способов автоматического выполнения сценария при за-

грузке будет организация на этот сценарий символической ссылки S39packetfilter (в этом случае сценарий будет выполняться перед S40network).

Маршрутизация, маскардинг, пересылка...

Этот документ посвящен фильтрации пакетов. Фильтр обеспечивает принятие решения о передаче пакета через систему или его отбрасывании. Однако Linux позволяет выполнять по отношению к пакетам не только фильтрацию, но и другие действия.

Одной из проблем является то, что ipchains используется также для управления маскардингом и прозрачными посредниками (проху), хотя эти механизмы в корне отличаются от фильтрации пакетов (текущие реализации Linux противоестественными средствами объединяют эти механизмы, создавая иллюзию тесной связи между ними).

Маскардингу и проху посвящены специальные документы HOWTO, а для управления автоматической пересылкой и пересылкой по портам используются специальные средства. Однако многие спрашивают об этих вещах, поэтому в данном документе описан ряд сценариев общего назначения и рассказано об их использовании. При рассмотрении этих сценариев вопросы безопасности не рассматриваются.

Как организовать маскардинг (Masquerading)

Предположим, что в качестве внешнего интерфейса Вашей системы используется ppp0¹. Приведенные ниже команды обеспечат маскардинг:

```
# ipchains -P forward DENY
# ipchains -A forward -i ppp0 -j MASQ
# echo 1 >
/proc/sys/net/ipv4/ip_forward
```

Средства настройки цепочек

Вы можете воспользоваться специальными программами² для настройки цепочек. Это хороший способ обеспечения безопасности системы на базе Linux – программа может управлять цепочками ipchains, а также работает с кодом брандмауэра в новых версиях ядра (2.4). Эту программу можно загрузить с сайта <http://www.watchguard.com>.

Установка параметров общего назначения для брандмауэра

Предположим, что у вас есть сайт littlecorp.com, связанный с внутренней сетью компании и имеющий одно коммутируемое соединение PPP с сетью Internet. Между сетью и внешним миром используется брандмауэр firewall.littlecorp.com с адресом 1.2.3.4. В локальной сети используется технология Ethernet, а ваш компьютер носит имя myhost.

В этом разделе описано несколько вариантов установки параметров общего назначения. Варианты имеют некоторые различия, поэтому рекомендуем внимательно прочесть описание всех способов.

¹ Для просмотра и задания параметров интерфейсов используется команда ifconfig.

² Примером такой программы может служить FireBox компании WatchGuard.

Частная сеть – традиционный посредник (Проху)

В этом сценарии пакеты из частной сети никогда не передаются в Internet и наоборот. IP-адреса частной сети выбираются из специальных групп адресов (10.*.*., 172.16.*.*-172.31.*.* или 192.168.*.*), выделенных для использования в частных сетях (см. RFC1918 - Address Allocation for Private Internets).

Для обеспечения взаимодействия между локальной сетью и Internet используется специальный компьютер (брандмауэр). На брандмауэре загружена специальная программа-посредник (проху), обеспечивающая обмен трафиком между сетями (FTP, web, telnet, RealAudio, новости Usenet, электронная почта и др.). Дополнительную информацию вы сможете найти в документе Firewall HOWTO.

Все службы Internet, к которым разрешен доступ из локальной сети, должны поддерживаться брандмауэром. Способы предоставления доступа в локальную сеть извне кратко описаны ниже (Ограниченный доступ к внутренним службам).

Пример: Организация доступа из частной сети к web-ресурсам Internet.

1. В частной сети используются адреса 192.168.1.*, мой хост имеет адрес 192.168.1.100, а интерфейс Ethernet на брандмауэре имеет адрес 192.168.1.1.
2. Web-прокси (например, squid) устанавливается на брандмауэре и работает на порту 8080 вы можете использовать другой порт).
3. Программа Netscape в частной сети настроена на работу с портом 8080 на брандмауэре в качестве прокси.
4. Сервер DNS в частной сети настраивать не нужно.
5. На брандмауэре должен быть настроен DNS.
6. Для частной сети не нужно указывать принятый по умолчанию шлюз (gateway).

Программа Netscape на моем хосте обращается по адресу <http://slashdot.org>.

1. Netscape подключается к порту 8080 на брандмауэре, используя порт 1050 на моем компьютере и запрашивая страницу <http://slashdot.org>.
2. Программа-посредник (проху) видит имя slashdot.org и открывает соединение с нужным адресом IP (используя порт 1025 на внешнем интерфейсе брандмауэра) и запрашивает у web-сервера (порт 80) искомую страницу.
3. Когда брандмауэр получает страницу через свое соединение с web-сервером, данные копируются в соединение брандмауэра с программой Netscape.
4. Netscape выводит на экран полученную страницу.

С точки зрения slashdot.org соединение организовано между адресом 1.2.3.4 (внешний PPP-интерфейс на брандмауэре), порт 1025 и адресом 207.218.152.131 (slashdot.org), порт 80. С точки зрения моего компьютера соединение организовано между адресом 192.168.1.100 (мой хост), порт 1050 и адресом 192.168.1.1 (интерфейс Ethernet на брандмауэре) порт 8080.

Частная сеть – прозрачный посредник

В этом сценарии пакеты из частной сети никогда не передаются в Internet и наоборот. IP-адреса частной сети должны выделяться на основе RFC1918 Address Allocation for Private Internets (т. е., 10.*.**, 172.16.*.*-172.31.*.* или 192.168.*.*).

Единственным способом связи с Internet из частной сети является подключение к брандмауэру, а последний является единственной машиной в сети, которая подключена к Internet. Вы загружаете на брандмауэре программу, называемую прозрачным посредником, для обмена данными между локальной сетью и внешним миром. Ядро будет передавать адресованные во внешнюю сеть пакеты программе-прокси вместо их передачи во внешнюю сеть.

Прозрачные посредники используются при работе с клиентскими программами, которые не должны знать о применении проху.

Все службы, которым обеспечивается доступ из Internet, должны устанавливаться на брандмауэре (см. параграф Ограниченный доступ к внутренним службам).

Пример: организация доступа в Internet из частной сети.

1. В частной сети используются адреса 192.168.1.*, мой хост имеет адрес 192.168.1.100, а интерфейс Ethernet на брандмауэре – адрес 192.168.1.1.
2. Прозрачный web-прокси (я полагаю, что есть заплатки для программы squid, позволяющие обеспечить это режим; если это не так, можете воспользоваться transпроху) установлен и настроен на брандмауэре для работы с портом 8080.
3. Ядро перенаправляет соединения с портом 80 на прокси, используя ipchains.
4. Программа Netscape в локальной сети настроена на прямое соединение (без прокси).
5. Для частной сети нужно организовать и настроить сервер DNS (вы можете организовать сервер DNS как прокси на брандмауэре).
6. Для передачи пакетов брандмауэру в частной сети должен быть указан используемый по умолчанию шлюз (gateway).

Программа Netscape на моем компьютере обращается к серверу <http://slashdot.org>.

1. Netscape запрашивает адрес для имени slashdot.org и после этого организует соединение через порт 1050, посылая запрос web-серверу (порт 80).
2. Когда пакеты от моего хоста (порт 1050) к серверу slashdot.org (порт 80) проходят через брандмауэр, они перенаправляются ожидающему прозрачному посреднику на порт 8080. Посредник открывает соединение (используя локальный порт 1025) с адресом 207.218.152.131 (порт 80), куда и были адресованы пакеты с моего хоста.
3. Когда проху получает страницу от web-сервера, данные копируются в соединение с Netscape.
4. Netscape выводит страницу на экран.

С точки зрения slashdot.org соединение организуется между адресом 1.2.3.4 (PPP-интерфейс на брандмауэре) через порт 1025 и 207.218.152.131 (slashdot.org), порт 80. С точки зрения моего хоста соединение организуется между 192.168.1.100 (мой хост), порт 1050 и 207.218.152.131 (slashdot.org), порт 80, хотя реально мой хост связан только с прозрачным прокси.

Частная сеть – маскардинг

В этом примере пакеты из частной сети никогда не передаются в Internet (и наоборот) без выполнения специальных действий. IP-адреса частной сети выделяются в соответствии с RFC1918 Address Allocation for Private Internets (т. е. 10.*.**, 172.16.*.*-172.31.*.* или 192.168.*.*).

Вместо использования программных посредников (проху) в этом случае применяются специальные функции ядра, называемые маскардингом (masquerading). Маскардинг обеспечивает изменение пакетов при их прохождении через брандмауэр таким образом, чтобы они выглядели исходящими непосредственно от брандмауэра. При передаче откликов на запросы брандмауэр модифицирует их так, чтобы они выглядели адресованными оригинальному получателю.

Маскардинг использует отдельные модули для обслуживания разных протоколов (таких, как FTP, RealAudio, Quake и т. п.). Для реально сложных в обслуживании протоколов могут использоваться средства автоматической пересылки (auto forwarding), позволяющие автоматически задавать рассылку по портам для связанного набора портов³.

Любые службы, к которым открыт доступ из Internet, должны быть реализованы на брандмауэре (см. параграф Ограниченный доступ к внутренним службам).

Пример: организация доступа в Internet из частной сети.

1. В частной сети используются адреса 192.168.1.*, мой хост имеет адрес 192.168.1.100, а порт Ethernet на брандмауэре - 192.168.1.1.
2. На брандмауэре организуется маскардинг для всех пакетов из частной сети, адресованных на порт 80 хостов Internet.
3. Netscape настраивается для работы с прямым соединением (без прокси).
4. Для частной сети нужно обеспечить сервер преобразования имен - DNS.
5. Для хостов частной сети в качестве принятого по умолчанию шлюза должен быть задан внутренний интерфейс брандмауэра.

Программа Netscape на моем хосте обращается к серверу <http://slashdot.org>.

1. Netscape получает имя slashdot.org и организует соединение по адресу этого сервера, используя локальный порт 1050 и передавая запросы в порт 80 сервера web.
2. Все пакеты от моего хоста (порт 1050) к slashdot.org (порт 80) передаются через брандмауэр и переписываются так, что выглядят исходящими от интерфейса PPP на брандмауэре с использованием порта 65000. Брандмауэр использует корректный адрес Internet (1.2.3.4), поэтому пакеты откликов от сервера slashdot.org корректно маршрутизируются в сети.
3. Когда пакет от slashdot.org (порт 80) приходит на брандмауэр firewall.littlecorp.com (порт 65000) они переписываются так, как будто были адресованы моему хосту (порт 1050). Магия маскардинга заключается в том, что брандмауэр, изменяя исходящие пакеты, всегда корректно меняет отклики на эти пакеты.
4. Netscape выводит полученную информацию на экран.

³ См. описание ipportfw' (ядра версии 2.0) или ipmasqadm' (ядра версии 2.1).

С точки зрения slashdot.org соединение организуется между адресом 1.2.3.4 (интерфейс PPP на брандмауэре), порт 65000 и адресом 207.218.152.131 (slashdot.org), порт 80. С точки зрения моего хоста соединение организуется между 192.168.1.100 (мой хост), порт 1050 и 207.218.152.131 (slashdot.org), порт 80.

Сеть общего пользования

В этом примере сеть компании является частью Internet – пакеты могут передаваться через границу сети без изменений. IP-адреса во внутренней сети выделяются из блока адресов, предоставленного компании, поэтому остальная часть сети Internet знает, как адресовать пакеты в вашу локальную сеть. В этом случае требуется постоянное подключение локальной сети к Internet.

В этом примере фильтрация пакетов используется для контроля за пакетами между локальной сетью и остальной частью Internet (прежде всего фильтры ограничивают доступ из Internet к ресурсам локальной сети).

Пример: организация доступа из частной сети к веб-ресурсам Internet.

1. В локальной сети используются адреса из зарегистрированного блока адресов IP (скажем, 1.2.3.*).
2. На брандмауэре разрешена передача всего трафика.
3. Программа Netscape настраивается на прямое соединение.
4. Для локальной сети должен быть указан сервер DNS.
5. Для частной сети в качестве принятого по умолчанию шлюза указывается внутренний интерфейс брандмауэра.

Netscape на моем компьютере обращается к серверу <http://slashdot.org>.

1. Netscape запрашивает имя slashdot.org и получает адрес сервера 207.218.152.131. После этого организуется соединение между локальным хостом с использованием локального порта 1050 и веб-сервером (порт 80) для загрузки веб-страницы.
2. Пакеты проходят через брандмауэр подобно прохождению их через другие маршрутизаторы между локальным хостом и сервером slashdot.org.

Цепочки IP Firewalling

В этом разделе приведены сведения, которые нужно знать для построения фильтра пакетов в соответствии с вашими задачами.

Как пакеты проходят через фильтры

Ядро начинает работать с тремя списками правил – эти списки называются цепочками брандмауэра или просто цепочками. Три стартовых цепочки называются input (вход), output (выход) и forward (пересылка). Когда пакет приходит (скажем, через порт Ethernet), ядро использует входную цепочку для принятия решения о судьбе пакета. Если пакет проходит через фильтр, ядро решает, куда этот пакет следует передать (это называется маршрутизацией (routing)). Если адресат расположен на другой машине, ядро использует цепочку

3. Netscape отображает на экране принятую страницу.

В этом случае организуется единственное соединение между 1.2.3.100 (мой хост), порт 1050 и 207.218.152.131 (slashdot.org), порт 80.

Ограниченный доступ к внутренним службам

В некоторых случаях разумней разрешить доступ из Internet к некоторым внутренним службам, нежели организовывать соответствующий сервис на брандмауэре. Такой доступ обеспечивается с помощью проху или маскардинга для внешних соединений.

Простейшим решением является использование редиректора, который представляет собой простую программу-посредник, которая ожидает соединения на данном порту. После организации такого соединения редиректор открывает соединение с внутренним хостом по определенному порту и копирует данные между этими соединениями. Примером может служить программа redir. Со стороны Internet соединение организуется с вашим брандмауэром, а со стороны локальной сети – с внутренним интерфейсом брандмауэра.

Другим способом ограничения доступа (он требует ядра 2.0 с заплатой ipportfw или 2.1 и выше) является использование рассылки по портам в ядре. В этом случае выполняются такие же действия, как при использовании программы redir, но выполняются они по другому – ядро переписывает проходящие через него пакеты, меняя адрес получателя и номер порта для того, чтобы указать на внутренний хост и порт. С точки зрения Internet соединение организуется с брандмауэром, а с точки зрения внутреннего сервера существует прямое соединение между хостом Internet и внутренним сервером.

Дополнительные сведения о маскардинге

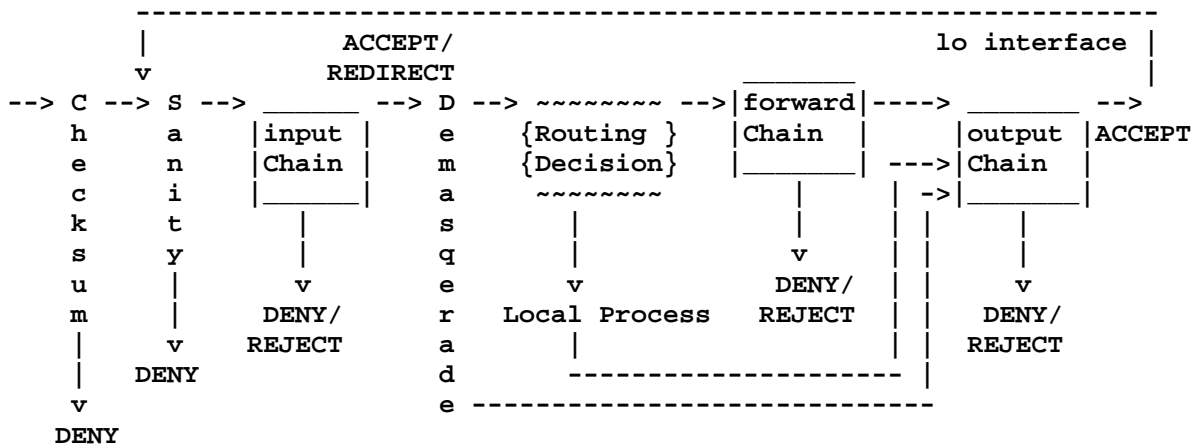
Дэвид Рэнч (David Ranch) написал превосходный документ HOWTO on Masquerading, который имеет множество пересечений с настоящим документом. Вы можете найти этот HOWTO по адресу <http://www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html>.

Официальную страницу, посвященную маскардингу, можно найти по адресу <http://ipmasq.cjb.net>.

forward. И, наконец, перед отправкой пакета адресату используется цепочка output.

Цепочка представляет собой список правил. Каждое из правил говорит: «если заголовок выглядит подобно xxx, пакет соответствует правилу». Если пакет не подходит под правило, он передается следующему правилу в цепочке. Когда будут использованы все правила в цепочке, ядро принимает решение о судьбе пакета в соответствии с назначением цепочки. В системах обеспечения безопасности правила обычно говорят ядру о необходимости отказа от пакета или отбрасывания пакета.

На рисунке показана блок-схема прохождения пакета через цепочку правил.



Ниже приведено краткое описание каждого из этапов прохождения пакета через брандмауэр:

Checksum: проверка целостности пакета по его контрольной сумме. Поврежденные пакеты отбрасываются.

Sanity: обычно «санитарная проверка» пакета осуществляется перед каждой цепочкой, но проверка на входе в систему имеет более важное значение, поскольку некоторые некорректные пакеты могут нарушать работу кода проверки правил. Некорректные пакеты отбрасываются (в этом случае делается запись в файл syslog и выводится сообщение на консоль).

input chain: это первая цепочка, которая используется брандмауэром для проверки пакета. Если в результате проверки не принимается решение DENY или REJECT, пакет передается дальше.

Demasquerade: если пакет является откликом на замаскированный пакет, выполняется операция демаскирования и пакет передается в выходную цепочку. Если IP Masquerading не используется, вы можете мысленно удалить это правило из схемы.

Routing decision: код маршрутизации проверяет адрес получателя пакета, после чего пакет передается на локальную обработку (см. *Local process*) или маршрутизируется на другую машину (см. *forward chain*).

Local process: процесс на локальной машине может принять пакет после этапа **Routing Decision** или передать его в выходную цепочку.

lo interface: если пакет от локального процесса адресован локальному процессу, он будет проходить через выходную цепочку интерфейса lo и возвращаться через входную цепочку интерфейса lo. Интерфейс lo обычно называют шлейфовым интерфейсом.

local: если пакет не создан локальным процессом, он передается в цепочку рассылки, в противном же случае пакет передается в выходную цепочку.

forward chain: эта цепочка используется для всех пакетов, которые проходят через брандмауэр на другую машину.

output chain: эта цепочка используется для всех пакетов, выводимых наружу.

Использование ipchains

Сначала следует проверить версию используемой программы ipchains:

```
$ ipchains --version
ipchains 1.3.9, 17-Mar-1999
```

Рекомендуется использовать ipchains 1.3.4 (она не использует длинных опций типа --sport) или 1.3.8 и выше – эти версии более стабильны.

ipchains имеет подробную документацию (используйте команду man ipchains) и для получения сведений об используемой версии вы можете посмотреть про-

граммный интерфейс (man 4 ipfw) или файл net/ipv4/ip_fw.c для ядер версии 2.1.x.

Есть также прекрасное справочное руководство Скотта Бронсона (Scott Bronson), распространяемое вместе с исходными текстами в формате PostScript(TM) для печати на бумаге A4 и US Letter.

Существует несколько различных операций с цепочками ipchains. Первая группа операций служит для управления цепочками в целом. Работа начинается с трех встроенных типов цепочек input (ввод), output (вывод) и forward (пересылка), которые нельзя удалить.

1. Создание новой цепочки (-N).
2. Удаление цепочки целиком (-X).
3. Изменение правил для предопределенной цепочки (-P).
4. Получение списка правил в цепочке (-L).
5. Удаление правила из цепочки (-F).
6. Сброс (обнуление) счетчиков пакетов и байтов для всех правил в цепочке (-Z).

Существует также несколько операций для управления правилами внутри цепочки:

1. Добавление нового правила в цепочку (-A).
2. Вставка правила в заданное место цепочки (-I).
3. Замена правила в указанной позиции цепочки (-R).
4. Удаление правила в указанной позиции цепочки (-D).
5. Удаление первого соответствующего правила в цепочке (-D).

Существует также несколько операций для маскардинга в цепочках ipchains:

1. Получение списка текущих соединений с маскардингом (-M -L).
2. Установка значения тайм-аута для маскардинга (-M -S)⁴. (But see "I can't set masquerading timeouts!").

Последняя (и, возможно, наиболее полезная) функция позволяет узнать, что произойдет с данным пакетом при прохождении его через цепочку.

Что будет происходить при загрузке операционной системы

Встроенные цепочки ipchains (input, forward и output) после инсталляции не задают каких-либо фильтров, т. е. каждая цепочка использует правило ACCEPT для всех пакетов. При установке ipchains будьте внимательны, в некоторых случаях программа автоматически загружается после инсталляции.

⁴ См. параграф Я не могу установить тайм-аут для маскардинга!

Операции с одним правилом

Манипуляция правилами – это основная и постоянная работа ipchains. Вы будете также использовать команды добавления (-A) и удаления (-D) правил. Остальные команды (-I для вставки и -R для замены) являются просто модификациями команд -A и -D.

Каждое правило задает набор условий, которым должен удовлетворять пакет, и содержит указание на действия, выполняемые по отношению к пакетам (target). Например, вы можете пожелать отбросить все пакеты ICMP, приходящие с адреса 127.0.0.1. В данном случае

```
# ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.2 ms
```

```
--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.2/0.2/0.2 ms
# ipchains -A input -s 127.0.0.1 -p icmp -j DENY
# ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

```
--- 127.0.0.1 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
#
```

Из приведенного выше примера можно видеть, что на первую команду ping приходит отклик (параметр -c 1 говорит о необходимости генерации единственного пакета).

После проверки добавим (-A) ко входной (input) цепочке правила, задающее для пакетов с адреса 127.0.0.1 (-s 127.0.0.1), содержащих протокол ICMP (-p ICMP) отказ от восприятия - DENY (-j DENY).

После добавления правила снова используем ping. Программа будет ждать отклика, который никогда не придет.

```
# ipchains -D input -s 127.0.0.1 -p icmp -j DENY
#
```

Синтаксис команды -D должен в точности совпадать с синтаксисом использованной для добавления команды -A (или -I или -R). При наличии в цепочке нескольких идентичных правил будет удалено только первое из них.

Спецификации фильтров

Опция -p служит для задания протокола, -s для указания отправителя пакетов, кроме того, существуют дополнительные опции для указания характеристик пакетов. Более подробное описание опций приведено ниже.

Задание IP-адресов отправителя и получателя

Адреса отправителя (-s) и получателя (-d) пакетов можно задать 4 способами. Наиболее общим является полное указание имени (например, localhost или www.linuxhq.com). Вторым способом является задание IP-адреса (например, 127.0.0.1). Третий и четвертый способ позволяют задать группу адресов (например, 199.95.207.0/24 или 199.95.207.0/255.255.255.0). Обе приведенных записи указывают на группу адресов с 199.95.207.0 по 199.95.207.255 включительно. Цифры после знака / говорят о числе битов в сетевой части адреса (маске подсети). /32 или /255.255.255.255 говорит о задании конкретного адреса, значение /0 указывает на любой адрес:

условиями является принадлежность к протоколу ICMP и отправка с адреса 127.0.0.1, а действием - DENY.

Адрес 127.0.0.1 задает loopback-интерфейс, который активен даже при отсутствии реальных соединений с сетью. Вы можете использовать программу ping для генерации пакетов ICMP (тип 8 - echo request), на которые все хосты должны давать отклик ICMP типа 0 (echo reply). Эта возможность полезна для тестирования.

Мы можем удалить это правило одним из 2 способов. Поскольку мы знаем, что это правило является единственным во входной цепочке, можно воспользоваться удалением правила по номеру:

```
# ipchains -D input 1
#
```

Второй способ является зеркальным отражением команды добавления правила (-A), реализуемым простой заменой -A на -D. Этот способ полезен для сложных цепочек, когда не хочется вычислять номер удаляемого правила в цепочке. Для удаления правила введите команду:

```
# ipchains -A input -s 0/0 -j DENY
#
```

Приведенное выше правило используется редко, поскольку оно просто аналогично отказу от задания адреса отправителя (опция -s не используется).

Обращение правила

Многие опции (включая -s и -d) позволяют обращать значение аргумента, обозначаемое знаком «!» (логическое отрицание). Например «-s ! localhost» обозначает все пакеты, кроме исходящих от localhost.

Не забывайте вставлять пробел между восклицательным знаком и аргументом.

Указание протокола

Для указания протокола служит флаг -p. Протокол можно задать по его номеру (если вы знаете номера субпротоколов IP) или по имени, используя зарезервированные обозначения TCP, UDP или ICMP. Строчные и прописные буквы при задании протокола не различаются, т. е. записи tcp и TCP эквивалентны.

Флаг протокола может использоваться с префиксом инверсии: -p ! TCP означает «все протоколы, за исключением TCP».

Задание портов UDP и TCP

В специальных случаях при задании протоколов TCP или UDP можно использовать дополнительный аргумент, задающий порт TCP/UDP или диапазон портов (см. параграф Обработка фрагментов). Диапазон задается с помощью символа «:» (например, диапазон 6000:6010 включает 11 портов от 6000 до 6010, включительно). Если значение нижней границы опущено, оно предполагается равным 0. Если опущена верхняя граница диапазона, ее значение принимается равным 65535. Для задания TCP-соединений с портами, номера которых меньше 1024, служит строка «-r TCP -s 0.0.0.0/0 :1023». Порты можно задавать также по именам (например, www).

Отметим, что спецификация портов может быть обращена с помощью знака !. так для задания пакетов TCP, не содержащих трафика WWW, можно использовать строку -r TCP -d 0.0.0.0/0 ! www

Важно понимать, что выражение «-r TCP -d ! 192.168.1.1 www» отличается от «-r TCP -d 192.168.1.1 ! www». Первая спецификация задает пакеты TCP, адресованные в порт WWW на любой машине, кроме 192.168.1.1. Вторая строка задает любые соединения TCP с любым портом на машине 192.168.1.1, за исключением порта WWW.

Приведем также пример спецификации, содержащей 2 инверсии – все порты, кроме WWW и любые адреса за исключением 192.168.1.1:

```
-r TCP -d ! 192.168.1.1 ! www
```

Задание типа и кода ICMP

Для протокола ICMP также можно использовать дополнительные аргументы, ICMP не использует портов (пакеты ICMP различаются по типу и коду), поэтому для задания дополнительных параметров используется другой синтаксис.

Вы можете указывать имена ICMP (используйте ipchains -h icmp для получения списка имен) после опции -s или задавать численные значения типа и кода – тип задается после опции -s, а код – после опции -d.

Протокол ICMP использует длинные имена – нет необходимости вводить их полностью, достаточно указать лишь часть имени, однозначно задающую тип.

Ниже приведен список имен наиболее часто используемых пакетов ICMP:

Номер	Имя	Используется
0	echo-reply	ping
3	destination-unreachable	Любой трафик TCP/UDP
5	redirect	Маршрутизация без использования демона
8	echo-request	ping
11	time-exceeded	traceroute

Отметим, что с именами ICMP нельзя использовать инверсию правил с помощью знака «!».

Не блокируйте все сообщения ICMP типа 3! (см. параграф Пакеты ICMP).

Задание интерфейса

Опция -i служит для указания имени интерфейса, для которого используется правило. Интерфейс является физическим устройством, через которое принимаются и передаются пакеты. Для получения списка имеющихся в системе интерфейсов можно воспользоваться

командой ifconfig. Активные интерфейсы помечены строкой **up**.

Интерфейс для входных пакетов (т. е. пакетов, проходящих через цепочку input), рассматривается как приемный интерфейс. Аналогично этому интерфейс для исходящих пакетов (пакетов, проходящих через цепочку output) рассматривается как передающий интерфейс. Интерфейс для пакетов, проходящих через цепочку forward, также рассматривается как выходной.

Можно создавать правила для интерфейсов, которые в настоящее время отсутствуют в системе – такие правила не будут действовать, пока соответствующий интерфейс не будет активизирован. Это очень удобно для коммутируемых соединений PPP и других соединений, которые не всегда активны.

Имя, заканчивающееся знаком «+» будет соответствовать всем интерфейсам, имена которых начинаются с заданной строки (например, eth+ будет задавать все активные интерфейсы Ethernet).

При задании интерфейсов можно использовать знак обращения «!» который позволяет выбрать пакеты, не соответствующие указанному интерфейсу.

Обработка пакетов TCP SYN

Иногда бывает полезно разрешить соединения TCP в одном направлении, запретив их в обратном. Например, вы можете разрешить соединения с внешними серверами WWW, но запретить соединения с вашей сетью с этих серверов. По наивности можно просто заблокировать пакеты TCP от таких серверов, но это закончится плохо. К сожалению, для TCP-соединений в процессе их использования требуется передача пакетов в обоих направлениях.

Для решения этой проблемы достаточно просто заблокировать пакеты, используемые для запроса соединения. Эти пакеты называются SYN-пакетами (более точно будет сказать, что это пакеты с установленным флагом SYN, но без флагов FIN и ACK). Запретив только эти пакеты, мы сможем заблокировать организацию соединений со стороны сервера, но при организации соединения со стороны локальной сети пакеты будут передаваться в обоих направлениях.

Запрет SYN-пакетов организуется с помощью флага -u, который можно использовать только для правил с указанным в них протоколом TCP. Например, для запрета соединений с адреса 192.168.1.1 можно использовать правило:

```
-r TCP -s 192.168.1.1 -u
```

Этот флаг также можно инвертировать с помощью знака ! – это приведет к запрету соединений со всех адресов, кроме указанного.

Обработка фрагментов

Иногда пакеты от приложений бывают слишком большими и для их передачи исходный пакет делится на несколько частей (фрагментов), которые передаются в форме отдельных пакетов. На приемной стороне фрагменты собираются заново в единый пакет.

Связанная с фрагментацией проблема состоит в том, что некоторые из используемых фильтрами параметров (в частности, адреса отправителя и получателя, тип и код ICMP, флаг TCP SYN) содержатся только в первом фрагменте большого пакета. Если компьютер имеет единственное соединение с внешней сетью, вы можете заставить ядро Linux собирать все проходящие через него фрагменты, задав при компиляции ядра для

опции IP «always defragment» значение Y. Такое решение почти полностью решает проблему.

С другой стороны важно понимать, как фрагменты трактуются правилами фильтрации пакетов. Если правило запрашивает отсутствующую в пакете информацию, это правило будет считаться невыполненным. В результате обработка первого фрагмента может отличаться от обработки остальных фрагментов. Например, правило -p TCP -s 192.168.1.1 www (задающее порт отправителя как www) никогда не будет выполняться для фрагментов отличных от первого.

Однако, вы можете задать специальные правила для второго и последующих фрагментов, используя -f. Обычно для фрагментов с номерами больше 1 задание таких параметров как порты TCP или UDP, тип и код ICMP или состояние флага TCP SYN является некорректной операцией.

```
# ipchains -A output -f -d 192.168.1.1 -j DENY
#
```

Результаты работы фильтров

Мы узнали о способах контроля за пакетами с помощью правил - осталось понять, что происходит в тех случаях, когда пакеты соответствуют правилам:

1. Значение счетчика байтов для правила увеличивается на размер пакета (с учетом заголовка).
2. Значение счетчика пакетов для правила увеличивается на 1.
3. Если правило задает протоколирование, в файл протокола заносится соответствующая запись.
4. Если правило задает это, изменяется значение поля Type Of Service (тип обслуживания пакета).
5. Если правило задает это, пакет маркируется (не поддерживается ядрами версии 2.0).
6. Проверяется заданное для правила действие с целью определения дальнейшей судьбы пакета.

Ниже эти действия будут более подробно рассмотрены в порядке их важности.

Задание действия (Target) для фильтра

Поле действия (target) говорит ядру, что делать с пакетами, соответствующими правилу. ipchains использует опцию -j (jump-to) для задания действия. Название действия должно иметь не более 8 букв; регистр символов имеет значение (действия RETURN и return являются различными).

Различия между ipchains и ipfwadm. Эта операция может применяться только для правил цепочки forward.

Действие REDIRECT заставляет ядро переслать пакет в локальный порт вместо его передачи адресату, указанному в заголовке. Эта опция может использоваться только для пакетов TCP или UDP. После опции -j REDIRECT может быть дополнительно указан порт (имя или номер), куда будет перенаправлен пакет независимо от его исходного назначения. Это действие применимо только к правилам цепочки input.

И, наконец, операция RETURN заставляет перейти в конец цепочки правил, пропустив проверку выполнения части правил в цепочке (см. параграф Установка политики для цепочек).

Для установки правил, не применимых ко второму и последующим фрагментам, можно использовать знак инверсии «!» перед флагом -f.

Обычно считают возможным пропускать через фильтр второй и последующие фрагменты, поскольку при отсутствии первого фрагмента сборка целого пакета невозможна. Однако в таких случаях система становится незащищенной против передачи потока фрагментов.

Примечание для сетевых администраторов: некорректные пакеты (пакеты TCP, UDP и ICMP, слишком короткие для того, чтобы код брандмауэра мог выяснить номер порта или код и тип ICMP) трактуются как фрагменты. Только фрагменты TCP, начинающиеся с позиции 8 явно отбрасываются брандмауэром (в таких ситуациях делается запись в syslog).

Например, следующее правило обеспечит отбрасывание всех фрагментов, принимаемых с адреса 192.168.1.1:

В простейшем случае для правила действие не задается. Этот тип правил (из часто называют правилами учета - accounting) полезен для учета пакетов того или иного типа. Независимо от выполнения такого правила ядро проверяет выполнение следующего правила в цепочке. Например, для учета числа пакетов с адреса 192.168.1.1 может служить правило:

```
# ipchains -A input -s 192.168.1.1
#
```

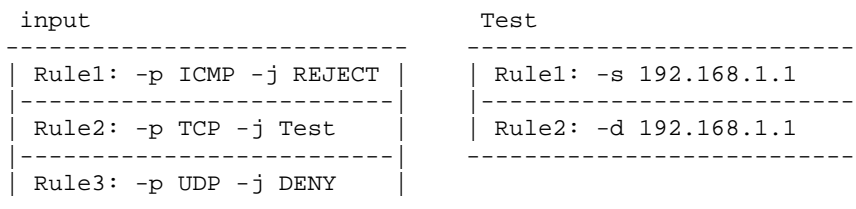
Для просмотра значений всех счетчиков, связанных с правилами, используйте команду ipchains -L -v.

Существует 6 predetermined действий для правил. Первые три - ACCEPT, REJECT и - DENY достаточно просты и понятны. ACCEPT обеспечивает дальнейшее прохождение пакета, DENY отбрасывает пакет (как будто его и не было), REJECT также отбрасывает пакет, но (если это не пакет ICMP) его отправителю передается ICMP-отклик, говорящий о недостижимости адресата (в дальнейшем мы будем использовать для этого действия термин ОТКАЗ).

Следующее действие - MASQ – заставляет ядро маскировать пакет. Для того, чтобы можно было использовать этот режим, ядро должно быть скомпилировано с опцией IP Masquerading. Детальную информацию о маскардинге вы сможете найти в документе Masquerading-HOWTO и приложении

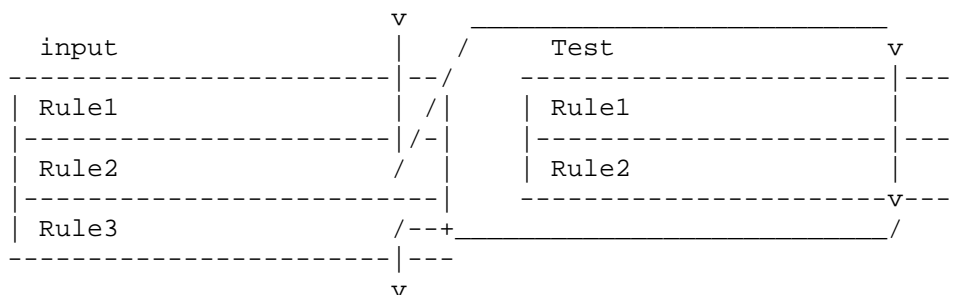
Любые другие действия могут применяться в определенных пользователем цепочках (см. параграф Операции с целой цепочкой). Проверка начинается с первого правила цепочки. Если это правило не позволяет определить судьбу пакета, последний передается следующему правилу и т. д. до окончания цепочки. Если ни одно из правил не определило судьбу пакета, вступают в действие правила следующей цепочки.

Пример прохождения пакета через цепочки показан на рисунке. Здесь используются две цепочки правил – predeterminedная цепочка input и пользовательская цепочка Test.



Рассмотрим пакет TCP, приходящий с адреса 192.168.1.1 на хост 1.2.3.4. Пакет попадает во входную цепочку и проверяется с помощью правила Rule1 – нет соответствия. Далее проверяется правило Rule2, действием которого является цепочка Test, поэтому пакет передается в цепочку Test, поскольку правило выполняется. Правило Rule1 в цепочке Test также выполня-

ется, но оно не задает действия, поэтому пакет передается на проверку правилу Rule2. Это правило не выполняется, но оно является последним в цепочке, поэтому происходит возврат к цепочке input. Правило Rule3 не выполняется и на этом проверка завершается. Путь пакета по цепочкам правил показан на рисунке:



Способы эффективного применения пользовательских цепочек описаны ниже (см. параграф Как организовать правила для брандмауэра).

Протоколирование (Logging) пакетов

Можно задать протоколирование фактов соответствия пакетов тому или иному правилу, задав для этого правило `Packet log: input DENY eth0 PROTO=17 192.168.2.1:53 192.168.1.1:1025 L=34 S=0x00 I=18 F=0x0000 T=254`

Эти записи в сжатой форме содержат информацию, которая может быть полезна специалистам, но мало что говорит обычному пользователю. Поля записи описаны ниже.

- input** задает имя цепочки, содержащей правило, которому соответствует пакет.
- DENY** действие, заданное правилом для пакета. Если в этом поле помещен символ «-», это говорит о том, что данное правило используется только для учета и не выполняет каких-либо операций над пакетами.
- eth0** указывает имя интерфейса, с которым связан пакет.
- PROTO=17** говорит о том, что пакет использует протокол 17 в соответствии с порядком следования записей в файле `/etc/protocols`. Чаще всего вам придется сталкиваться с протоколами 1 (ICMP), 6 (TCP) и 17 (UDP).
- 192.168.2.1** – адрес отправителя пакета.
- :53** – номер порта, через который был отправлен пакет. Список портов можно найти в файле `/etc/services`. Порт 53 используется для обмена информацией с серверами имен (DNS). Для пакетов ICMP это поле задает тип пакета, а для остальных протоколов используется значение 65535.
- 192.168.1.1** – адрес получателя пакета.
- :1025** – номер порта, в который адресован пакет (UDP и TCP), тип пакета ICMP или 65535 для остальных протоколов.
- L=34** – общая длина пакета в байтах.

вила флаг `-l`. Обычно протоколирование не используется для широко распространенных пакетов, но может оказаться полезным для протоколирования исключительных ситуаций.

Ядро протоколирует ситуации выполнения правил с флагом `-l`, делая записи типа:

- S=0x00** задает значение поля Type of Service (тип обслуживания) – типы обслуживания, используемые ipchains, получаются в результате деления этого числа на 4.
- I=18** – идентификатор протокола IP.
- F=0x0000** – 16-битовое смещение фрагмента и флаги. Значения, начинающиеся с 0x4 или 0x5, говорят об установке флага Don't Fragment (не фрагментировать). 0x2 и 0x3 говорят о наличии других фрагментов (флаг More Fragments), остальные значения задают смещение фрагмента, поделенное на 8.
- T=254** – значение поля Time To Live (время жизни) пакета. Это поле уменьшается на 1 каждым маршрутизатором на пути доставки пакета и обычно имеет значение от 15 до 255.
- (#5)** – номер правила, которое связано с записью в файле протокола (в последних версиях ядра могут использоваться квадратные скобки []).

В стандартных системах Linux выдаваемая ядром протокольная информация захватывается демоном `klogd` (the kernel logging daemon), который передает их демону `syslogd` (the system logging daemon). Файл `/etc/syslog.conf` задает параметры конфигурации `syslogd`, указывая назначение для каждого объекта (facility) и уровня (level) – в нашем случае объектом является ядро, а в качестве уровня используется `info`. Например, в моей системе (Debian) файл `/etc/syslog.conf` содержит две строки для пары `kern.info`:

```

kern.*                                -/var/log/kern.log
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none                    -/var/log/messages
  
```

Это означает, что сообщения записываются в файл /var/log/kern.log и дублируются в /var/log/messages. Более детальные сведения о протоколировании можно получить с помощью команды man syslog.conf.

Управление типом обслуживания (TOS)

В заголовке пакетов IP присутствуют 4 бита, позволяющие задать для пакета тип обслуживания (Type of Service или TOS). Эти биты определяют опции обработки пакета - Minimum Delay (минимальная задержка), Maximum Throughput (максимальная пропускная способность), Maximum Reliability (максимальная надежность) и Minimum Cost (минимальная стоимость). Для каждого пакета может быть установлен только один из этих битов.

Rob van Nieuwkerk – автор кода обработки TOS, говорит следующее: «Особенно важна для меня минимальная задержка. Я устанавливаю этот флаг для «интерак-

```
ipchains -A output -p tcp -d 0.0.0.0/0 telnet -t 0x01 0x10
ipchains -A output -p tcp -d 0.0.0.0/0 ftp -t 0x01 0x10
ipchains -A output -p tcp -s 0.0.0.0/0 ftp-data -t 0x01 0x08
```

Флаг -t поддерживает два дополнительных параметра, задаваемых 16-ричными числами. Для первого параметра выполняется операция AND с текущим значени-

тивного» трафика на внешнем интерфейсе моего Linux-маршрутизатора, подключенного к модему 33,6 кбит/с. Linux распределяет пакеты в 3 очереди с разными приоритетами. Таким образом можно обеспечить одновременную поддержку чувствительного к задержке трафика и перекачку больших файлов.»

Примечание: обычно у вас нет возможности контроля принимаемых пакетов и вы можете устанавливать флаги приоритета только для исходящего трафика. Для согласования приоритетов с удаленной стороной должен использоваться протокол типа RSVP (я ничего не знаю о нем, не спрашивайте).

Наиболее распространенным способом использования флагов является установка бита Minimum Delay для telnet и управления ftp, а для передачи данных по протоколу FTP – установка бита Maximum Throughput. Сделать эти установки можно с помощью приведенных здесь правил:

ем поля TOS, а для второго – операция XOR с полем TOS.

Если это слишком сложно для вас, воспользуйтесь приведенной таблицей:

Название типа обслуживания (TOS)	Значение	Типичное использование
Minimum Delay (минимальная задержка)	0x01 0x10	ftp, telnet
Maximum Throughput (максимальное пропускание)	0x01 0x08	ftp-data
Maximum Reliability (максимальная надежность)	0x01 0x04	snmp
Minimum Cost (минимальная стоимость)	0x01 0x02	nntp

По словам Энди Клина (Andi Kleen): «Может оказаться полезным добавить ссылку на параметр txqueuelen при обсуждении битов TOS в контексте ipchains. Размер очереди в принятом по умолчанию устройстве подобранный для адаптеров ethernet, слишком велик для модемов и позволяет спланировать 3 очереди на модемном канале в соответствии со значением TOS. Хорошей идеей будет установка значения 4-10 для модемного канала или одного b-канала ISDN. Для пакетных устройств требуется более длинная очередь. Эта проблема проявляется в ядрах версий 2.0 и 2.1, а в ядре 2.1 используется флаг ifconfig (с последней версией nettools). Для ядер 2.0 требуется наложение заплат на код драйверов устройств».

Таким образом, для максимальной реализации преимуществ TOS для модемных каналов PPP следует включить строку ifconfig \$1 txqueuelen в ваш файл /etc/ppp/ip-up. Используемое в этой строке число зависит от скорости модема и размера буфера в модеме. Воспользуемся еще раз рекомендациями Энди Клина: «Наилучшее значение для каждой конфигурации определяется экспериментальным путем. Если очередь маршрутизатора слишком мала, пакеты будут теряться. Переписывание TOS позволяет воспользоваться преимуществами приоритизации даже для некооперативных программ (отметим, что все стандартные программы Linux являются кооперативными).

Маркировка пакетов

Маркировка обеспечивает комплексное и эффективное взаимодействие с новой реализацией QoS Алексея Кузнецова, а также с рассылкой на основе маркеров в последних ядрах серии 2.1. Эта опция совсем не поддерживается ядрами серии 2.0.

Операции с целой цепочкой

Очень полезной функцией ipchains является возможность группировки связанных правил в цепочки. Вы можете как угодно называть свои цепочки, не используя лишь predetermined имен цепочек (input, output forward) и действий (MASQ, REDIRECT, ACCEPT, DENY, REJECT и RETURN). Я рекомендую избегать записи имен цепочек заглавными буквами, поскольку это может привести к конфликтам с будущими версиями. Длина имени цепочки может достигать 8 символов.

Создание новой цепочки

Для создания новой цепочки с именем test достаточно использовать приведенную ниже строку:

```
# ipchains -N test
#
```

Создав цепочку, вы можете включить в нее правила в соответствии с приведенными ниже рекомендациями.

Удаление цепочки

Для удаления пользовательской цепочки служит приведенная ниже строка:

```
# ipchains -X test
#
```

При удалении цепочек имеются некоторые ограничения – удаляемая цепочка должна быть пустой (см. параграф Уничтожение (Flushing) цепочки) и не должна использоваться в качестве действия в каком-либо из оставшихся правил. Удаление встроенных цепочек не поддерживается.

Уничтожение (Flushing) цепочки

Для удаления всех правил из цепочки используется команда -F.

```
# ipchains -F forward
#

# ipchains -L input
Chain input (refcnt = 1): (policy ACCEPT)
target    prot opt    source    destination    ports
ACCEPT   icmp ----- anywhere
# ipchains -L test
Chain test (refcnt = 0):
target    prot opt    source    destination    ports
DENY     icmp ----- localnet/24
#
```

Значение refcnt показывает число правил в цепочке. Это поле должно иметь нулевое значение (пустая цепочка) для того, чтобы цепочку можно было удалить. Если в команде не указать имя цепочки, будет выведен список правил для всех цепочек (даже пустых).

Для команды -L поддерживаются три опции. Опция -n (numeric) предотвращает при просмотре преобразова-

```
# ipchains -v -L input
Chain input (refcnt = 1): (policy ACCEPT)
pkts bytes target prot opt  tosa tosx ifname mark source destination ports
10  840  ACCEPT icmp ----- 0xFF 0x00 lo          anywhere anywhere any
```

Отметим, что при выводе значений счетчиков используются суффиксы К (1000), М (1000000) или G (1000000000). Для вывода показаний счетчиков в полном формате (без использования суффиксов) служит флаг -x (expand numbers).

```
# ipchains -v -L input
Chain input (refcnt = 1): (policy ACCEPT)
pkts bytes target prot opt  tosa tosx ifname mark source destination ports
10  840  ACCEPT icmp ----- 0xFF 0x00 lo          anywhere anywhere any
# ipchains -Z input
# ipchains -v -L input
Chain input (refcnt = 1): (policy ACCEPT)
pkts bytes target prot opt  tosa tosx ifname mark source destination ports
0    0  ACCEPT icmp ----- 0xFF 0x00 lo          anywhere anywhere any
#
```

Иногда перед сбросом счетчиков нужно узнать их показания. В приведенном выше примере сначала используется команда -L для просмотра, а потом -Z для сброса счетчиков. Однако можно задать эти команду

```
# ipchains -L -v -Z
Chain input (policy ACCEPT):
pkts bytes target prot opt  tosa tosx ifname mark source destination ports
10  840  ACCEPT icmp ----- 0xFF 0x00 lo          anywhere anywhere any
Chain forward (refcnt = 1): (policy ACCEPT)
Chain output (refcnt = 1): (policy ACCEPT)
Chain test (refcnt = 0):
0    0  DENY   icmp ----- 0xFF 0x00 ppp0          localnet/24 anywhere any
# ipchains -L -v
Chain input (policy ACCEPT):
pkts bytes target prot opt  tosa tosx ifname mark source destination ports
10  840  ACCEPT icmp ----- 0xFF 0x00 lo          anywhere anywhere any
Chain forward (refcnt = 1): (policy ACCEPT)
Chain output (refcnt = 1): (policy ACCEPT)
Chain test (refcnt = 0):
0    0  DENY   icmp ----- 0xFF 0x00 ppp0          localnet/24 anywhere any
#
```

Установка политики для цепочек

Мы рассматривали выше судьбу пакетов по достижении ими конца цепочки (см. параграф Задание действия (Target) для фильтра). В таких случаях оконча-

Если имя цепочки не указано в качестве параметра, правила удаляются из всех цепочек.

Просмотр списка цепочек

Вы можете вывести на экран список всех правил цепочки с помощью команды -L.

```
destination    ports
anywhere       any

destination    ports
anywhere       any
```

ние адресов в имена с помощью сервера DNS – это обеспечивает более быстрый вывод. Эта опция также обеспечивает вывод номеров портов взамен имен. Опция -v' показывает все детали правил (счетчики пакетов и байтов, маски TOS, интерфейсы и маркировку пакетов):

Сброс (обнуление) счетчиков

Иногда может потребоваться сброс показаний счетчиков. Для решения этой задачи служит команда -Z (zero counters):

одновременно, как показано ниже. К сожалению, в таких случаях нельзя ограничиться одной цепочкой – команда выводит на экран и сбрасывает в 0 показания счетчиков для всех цепочек.

тельную судьбу пакета определяет политика цепочки. Политика поддерживается только для встроенных цепочек, поэтому при завершении проверки правил в пользовательской цепочке пакет возвращается в ту цепочку, из которой он был направлен в пользовательскую.

В качестве политики встроенных цепочек могут использоваться 4 predetermined варианта - ACCEPT, DENY, REJECT или MASQ. Политика MASQ может использоваться только для цепочки forward.

Отметим также, что действие RETURN в правиле встроенной цепочки приводит к реализации политики цепочки в случае выполнения этого правила.

Операции с маскардингом

Существует несколько операций, с помощью которых можно воздействовать на IP Masquerading. Эти операции включены в ipchains поскольку неразумно было создавать для этих целей отдельный инструмент (впоследствии эта ситуация может измениться).

Для IP Masquerading используется команда -M, которую можно объединять с командой -L для просмотра маскируемых соединений или с командой -S для установки параметров маскардинга.

Вместе с командой -L можно также использовать опцию -n (показывать адреса и номера портов взамен имен) или -v (показывать детали для маскируемых соединений).

Команда -S должна использоваться с 3 параметрами, задающими тайм-ауты (в секундах) - для сеансов TCP, для сеансов TCP после пакета FIN и для пакетов UDP. Если вы не хотите менять какой-либо из этих параметров, просто задайте для него значение 0. Принятые по умолчанию значения тайм-аутов приведены в файле /usr/src/linux/include/net/ip_masq.h (обычно 15 минут, 2 минуты и 5 минут, соответственно).

Достаточно часто значение первого тайм-аута меняют для протокола FTP (см. параграф Проблемы с FTP).

```
# ipchains -C input -p tcp -y -i eth0 -s 192.168.1.1 60000 -d 192.168.1.2 www
packet accepted
#
```

Работа с несколькими правилами сразу и определение результата

Иногда одна строка команды может воздействовать сразу на несколько правил. Это может происходить в двух случаях. Первый вариант возникает в тех случаях, когда имя хоста преобразуется (с помощью сервера DNS) в несколько адресов IP. Ipchains в таких случаях будет выполнять заданные операции по отношению к каждому из определенных адресов. Если имя www.foo.com преобразуется в три адреса IP, а www.bar.com - в 2, команда ipchains -A input -j reject -s www.bar.com -d www.foo.com будет включать в цепочку input сразу 6 правил.

Другим способом заставить ipchains выполнять множество операций является использование флага двунаправленности -b. При установке этого флага ipchains ведет себя так, как будто команда была введена дважды, при этом во втором случае значения аргументов -s и -d меняются местами (отправитель становится полу-

```
# ipchains -v -b -C input -p tcp -f -s 192.168.1.1 -d 192.168.1.2 -i lo
tcp opt ---f- tos 0xFF 0x00 via lo 192.168.1.1 -> 192.168.1.2 * -> * packet accepted
tcp opt ---f- tos 0xFF 0x00 via lo 192.168.1.2 -> 192.168.1.1 * -> * packet accepted
#
```

Полезные примеры

У меня есть коммутируемое соединение PPP (-i ppp0). Всякий раз при организации соединения я собираю новости (-p TCP -s news.virtual.net.au nntp) и почту (-p TCP -s mail.virtual.net.au pop-3). Кроме того, используется протокол FTP для обновления программ с сайта Debian (-p TCP -y -s ftp.debian.org.au ftp-data). Для ра-

Проблемы, которые могут встретиться при изменении значений тайм-аута, описаны ниже (см. параграф Я не могу установить тайм-аут для маскардинга!).

Проверка пакета

Иногда может потребоваться проверить, что произойдет с пакетом в результате прохождения через фильтры (например, при настройке брандмауэра). Ipchains позволяет проводить такие с помощью команды -C - в этом случае выполняются все проверки, как это делало бы ядро.

Вы задаете цепочку, используемую для тестирования, указывая имя цепочки вместе с командой -C. При тестировании вы можете выбрать любую цепочку, не обращая внимания на порядок проверки цепочек ядром.

Детали пакета задаются с использованием такого же синтаксиса, как при установке правил брандмауэра. В частности, вы можете использовать опции -p (протокол), -s (отправитель), -d (получатель) и -i (интерфейс). Для протоколов TCP или UDP можно также указать порт для отправителя и получателя (указание диапазона портов в этом случае не поддерживается), а для ICMP - тип и код (если не используется ключ -f. Если проверка выполняется для протокола TCP и флаг -f не задан, можно установить флаг -y, позволяющий проверить пакеты с установленным битом SYN.

Ниже приведен пример команды для проверки пакета TCP SYN от 192.168.1.1, порт 60000, адресованного в порт 192.168.1.2: www и приходящего на интерфейс eth0, с помощью цепочки input (организация соединения с сервером WWW).

чателем и наоборот). Для того, чтобы избежать пересылки по адресу 192.168.1.1 или с этого адреса, можно использовать команду:

```
# ipchains -b -A forward -j reject -s
192.168.1.1
#
```

Мне не нравится опция -b и для удобства я рекомендую использовать команду ipchains-save, описанную ниже. Опция -b может использоваться с командами вставки -I, удаления -D (исключая варианты этой команды с указанием номера правила), присоединения -A и проверки правил -C.

Другой полезной опцией является -v (verbose), которая позволяет вывести на экран точную информацию о том, что произойдет в результате выполнения вашей команды. Особенно полезна эта опция при использовании описанных выше команд, работающих с множеством правил. В приведенном ниже примере проверяется поведение фрагментов, передаваемых между адресами 192.168.1.1 и 192.168.1.2.

боты с web-серверами используется прокси-сервер провайдера (-p TCP -d proxy.virtual.net.au 8080), но при этом требуется подавлять рекламу с doubleclick.net при работе с файловым архивом Dilbert (-p TCP -y -d 199.95.207.0/24 and -p TCP -y -d 199.95.208.0/24).

Я не принимаю во внимание попытки доступа к моей машине по протоколу ftp во время сеансов доступа в Internet (-p TCP -d \$LOCALIP ftp), но я не хочу, чтобы кто-нибудь извне видел IP-адреса моей внутренней

сети (-s 192.168.1.0/24). Такое решение обычно называют обманкой (IP spoofing) – его реализация для ядер версии 2.1.x и выше описана в параграфе Как организовать защиту от обманок IP Spoof?.

```
# ipchains -A output -d 199.95.207.0/24 -j REJECT
# ipchains -A output -d 199.95.208.0/24 -j REJECT
#
```

После этого я хочу установить приоритеты для различных исходящих пакетов (в моем случае нет смысла делать это для входящих пакетов). Поскольку число правил невелико, их можно поместить в одну цепочку, назвав ее ppp-out.

```
# ipchains -N ppp-out
# ipchains -A output -i ppp0 -j ppp-out
#
```

Минимальная задержка для web и telnet.

```
# ipchains -A ppp-out -p TCP -d proxy.virtual.net.au 8080 -t 0x01 0x10
# ipchains -A ppp-out -p TCP -d 0.0.0.0/0 telnet -t 0x01 0x10
#
```

Путь с минимальной стоимостью для ftp data, nntp, pop-3:

```
# ipchains -A ppp-out -p TCP -d 0.0.0.0/0 ftp-data -t 0x01 0x02
# ipchains -A ppp-out -p TCP -d 0.0.0.0/0 nntp -t 0x01 0x02
# ipchains -A ppp-out -p TCP -d 0.0.0.0/0 pop-3 -t 0x01 0x02
#
```

Существуют также некоторые ограничения для пакетов, принимаемых через интерфейс ppp0 – поместим эти правила в цепочку ppp-in:

```
# ipchains -N ppp-in
# ipchains -A input -i ppp0 -j ppp-in
#
```

Принимаемые от интерфейса ppp0 пакеты не должны быть отправлены из сети 192.168.1.*:

```
# ipchains -A ppp-in -s 192.168.1.0/24 -l -j DENY
#
```

Я разрешаю прием пакетов UDP только для DNS (используется кэширующий сервер имен, который пересылает все запросы по адресу 203.29.16.1, поэтому я ожидаю откликов только с этого адреса), входящего

```
# ipchains -A ppp-in -p UDP -s 203.29.16.1 -d $LOCALIP dns -j ACCEPT
# ipchains -A ppp-in -p TCP -s 0.0.0.0/0 ftp-data -d $LOCALIP 1024:5999 -j ACCEPT
# ipchains -A ppp-in -p TCP -s 0.0.0.0/0 ftp-data -d $LOCALIP 6010: -j ACCEPT
# ipchains -A ppp-in -p TCP -d $LOCALIP ftp -j ACCEPT
#
```

Я разрешаю принимать отклики TCP

```
# ipchains -A ppp-in -p TCP ! -y -j ACCEPT
#
```

И, наконец, локальный обмен пакетами ничем не ограничивается:

```
# ipchains -A input -i lo -j ACCEPT
#
```

По умолчанию для цепочки input используется действие DENY (отказ), поэтому, все остальные пакеты просто отбрасываются:

```
# ipchains -P input DENY
#
```

Примечание: Порядок правил в цепочке не будет соответствовать рассмотренному выше порядку. Безопаснее будет сначала установить правило DENY и после него вставить остальные правила. Если ваши правила требуют обращения к серверу DNS для разрешения имен, нужно сначала открыть доступ к серверу имен.

Использование ipchains-save

После установки правил для брандмауэра нет необходимости вводить их заново или набивать вручную файл сценария. Команда ipchains-save позволяет сохранить в файле установленные в настоящее время цепочки проверок.

ipchains-save может сохранять одну указанную цепочку или все цепочки сразу (если имя не указано). Эта команда поддерживает единственную опцию -v, которая обеспечивает вывод (на stderr) сохраняемых в фай-

Реализация описанной схемы достаточно проста, поскольку во внутренней сети просто нет других машин. Я не хочу, чтобы какой-либо локальный процесс (например, Netscape, lynx и т. п.) связывался с doubleclick.net:

ле правил. Для цепочек input, output и forward обеспечивается также запись в файл политики цепочки.

```
# ipchains-save > my_firewall
Saving `input'.
Saving `output'.
Saving `forward'.
Saving `ppp-in'.
Saving `ppp-out'.
#
```

Использование ipchains-restore

Команда ipchains-restore позволяет восстановить цепочки, сохраненные ранее с помощью ipchains-save. Эта команда поддерживает 2 опции: -v описывает каждое добавляемое правило, а -f очищает пользовательские цепочки, если они были заданы ранее (см. ниже). При обнаружении пользовательских цепочек в input команда ipchains-restore проверяет каждую добавляемую цепочку (не была ли она задана ранее). При обна-

ружении таких цепочек на экран выдается запрос на ее отбрасывание (удаление всех правил) или сохранение.

```
# ipchains-restore < my_firewall
Restoring `input'.
Restoring `output'.
Restoring `forward'.
Restoring `ppp-in'.
Chain `ppp-in' already exists. Skip or flush? [S/f]? s
Skipping `ppp-in'.
Restoring `ppp-out'.
Chain `ppp-out' already exists. Skip or flush? [S/f]? f
Flushing `ppp-out'.
#
```

Разное.

В этом разделе приведены ответы на наиболее распространенные вопросы и содержится информация, не включенная в предыдущие разделы.

Как организовать правила для брандмауэра

Здесь возможно несколько подходов. Например, вы можете оптимизировать скорость работы (для наиболее распространенных пакетов выполняется мини-

```
# Re-create the `ppp-in' chain.
ipchains-restore -f < ppp-in.firewall
```

```
# Replace DENY rule with jump to ppp-handling chain.
ipchains -R input 1 -i ppp0 -j ppp-in
```

В сценарии разрыва соединения снова устанавливается отказ для всех пакетов во входной цепочке:

```
ipchains -R input 1 -i ppp0 -j DENY
```

Что не следует фильтровать

Существует несколько важных моментов, которые следует принимать во внимание до активизации каких-либо фильтров.

Пакеты ICMP

Пакеты ICMP используются (наряду с другими) для индикации сбоев другим протоколам (таким, как TCP и UDP). Пакеты destination-unreachable (адресат недоступен) являются примером таких индикаторов. Блокирование этих пакетов может привести к тому, что вы просто не будете получать сообщений Host unreachable или No route to host и приложения будут ожидать до бесконечности отклика от недоступных серверов. Это не фатально, но достаточно неудобно.

Более важное значение имеет роль пакетов ICMP в работе механизма MTU discovery. Все качественные реализации TCP (включая Linux) используют механизм MTU discovery для определения максимального размера пакетов, которые могут быть переданы адресату без фрагментации (фрагментация снижает производительность, особенно при потере отдельных фрагментов). Механизм MTU discovery основан на передаче пакетов с установленным битом Don't Fragment (не фрагментировать). Если в ответ на такой пакет приходит отклик Fragmentation needed but DF set (требуется фрагментация), размер пакета автоматически уменьшается. Пакеты Fragmentation needed but DF set являются разновидностью откликов destination-unreachable и при их фильтрации локальный хост не будет снижать значения MTU, что с неизбежностью приведет к фрагментации пакетов и снижению скорости обмена. От-

Если вы задали опцию -f в командной строке, запросов не выдается и все цепочки отбрасываются.

Например:

мальное число проверок) или уделить больше внимания управляемости системы.

Если вы используете коммутируемое соединение скажем, PPP), вы можете захотеть установить полный запрет для входной цепочки при старте системы (-i ppp0 -j DENY) и запускать рабочие правила из сценария организации соединения, подобно приведенному ниже примеру:

метим, что в общем случае блокируются все пакеты ICMP redirect (тип 5), которые можно использовать для манипуляции с маршрутами (хотя в хороших стеках IP против этого приняты специальные меры).

TCP-соединение с DNS (сервер имен)

Если вы планируете блокировать исходящие соединения TCP, помните, что DNS не всегда использует UDP – если размер отклика от сервера превышает 512 байтов, клиент использует TCP-соединение (оно по прежнему адресовано в порт 53) для получения данных. При запрете таких соединений у вас могут возникнуть проблемы доступа к серверам DNS (слишком большие задержки и даже отказы).

Если ваши DNS-запросы всегда используют один внешний адрес (прямое указание имени сервера в файле /etc/resolv.conf или использование кэширующего сервера в режиме пересылки - forward), вы можете специально разрешить соединения TCP для этого адреса (с портом domain при использовании кэширующего сервера или портом > 1023 при использовании /etc/resolv.conf).

Проблемы с FTP

Классической проблемой для фильтрации пакетов является протокол FTP. Этот протокол использует два режима - активный (традиционный) и пассивный (более поздняя реализация). Web-браузеры обычно по умолчанию работают в пассивном режиме, а консольные клиенты FTP – в активном.

В активном режиме удаленная сторона, желая передать файл (или даже результат выполнения команды ls или dir) пытается организовать соединение TCP с локальной машиной. Это означает, что вы не можете фильтровать такие соединения, не нарушая работы активных клиентов FTP.

При использовании пассивного режима такой проблемы не возникает – соединение организуется со стороны клиента даже для входящих пакетов.

С другой стороны, рекомендуется разрешать соединения TCP только для портов с номерами более 1024 за исключением диапазона 6000 - 6010 (6000 используется для X-Windows).

Фильтрация Ping of Death

Linux-системы сегодня устойчивы к знаменитой атаке Ping of Death, которая основано на передаче слишком больших пакетов ICMP, которые приводили к переполнению буферов стека TCP на приемной стороне (havoc).

Если в сети за брандмауэром сохранились машины, чувствительные к таким атакам, вы можете просто заблокировать фрагменты ICMP. Нормальные пакеты ICMP не требуют фрагментации, поэтому такая операция не будет мешать нормальной работе системы. Приходилось слышать (эта информация не имеет подтверждений) о том, что работа некоторых систем может нарушаться в результате приема лишь последнего фрагмента слишком большого пакета ICMP, поэтому не следует ограничиваться блокировкой лишь первого фрагмента.

Блокирование фрагментов ICMP на деле является лишь временным решением проблемы, поскольку нет причин, запрещающих использовать для подобных атак фрагменты TCP или UDP (или неизвестных протоколов).

Фильтрация Teardrop и Bonk

Teardrop и Bonk представляют собой два типа атаки (в основном против машин Microsoft Windows NT), которые используют перекрывающиеся фрагменты. Для предотвращения таких атак следует организовать дефрагментацию на маршрутизаторе Linux или запретить передачу любых фрагментов на неустойчивые к таким атакам компьютеры.

Фильтрация Fragment Bomb

В некоторых недостаточно устойчивых реализациях стеков TCP могут возникать проблемы при работе с большим числом фрагментов пакета, пока не получены все фрагменты. В Linux таких проблем не возникает. Вы можете фильтровать такие фрагменты или скомпилировать ядро с опцией IP: always defragment = Y (если

```
# This is the best method: turn on Source Address Verification and get
# spoof protection on all current and future interfaces.
if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ]; then
    echo -n "Setting up IP spoofing protection..."
    for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
        echo 1 > $f
    done
    echo "done."
else
    echo PROBLEMS SETTING UP IP SPOOFING PROTECTION. BE WORRIED.
    echo "CONTROL-D will exit from this shell and continue system startup."
    echo
# Start a single user shell on the console
/sbin/sulogin $CONSOLE
fi
```

Если вам не удастся изменить сценарий загрузки, можно вручную вставить правила для защиты каждого интерфейса (это потребует понимания работы интерфейсов). Ядра версии 2.1 автоматически отбрасывают

ваш маршрутизатор Linux будет способен маршрутизировать собранные пакеты).

Изменение правил брандмауэра

Существует ряд аспектов, которые следует принимать во внимание при смене правил брандмауэра. При неаккуратной работе в процессе замены правил в системе могут образоваться бреши. Простейшим способом безопасной замены правил является использование приведенных ниже команд в указанном порядке:

```
# ipchains -I input 1 -j DENY
# ipchains -I output 1 -j DENY
# ipchains -I forward 1 -j DENY
```

... внесение изменений ...

```
# ipchains -D input 1
# ipchains -D output 1
# ipchains -D forward 1
#
```

В результате использования такого варианта на время замены правил будет обеспечено отбрасывание всех пакетов.

Если вносимые изменения ограничиваются одной цепочкой, можно создать новую цепочку с новым правилом и применить опцию -R для вставки нового правила, после чего старое правило можно удалить из цепочки. Такая замена происходит быстро и обеспечивает безопасность.

Как организовать защиту от обманок IP Spoof?

IP spoofing представляет собой метод передачи хостом пакетов, которые содержат в качестве адреса отправителя чужой адрес. Поскольку часто используется фильтрация пакетов по адресам отправителей, этот метод позволяет обойти простые пакетные фильтры. Обманки также позволяют скрыть источник атаки при использовании атак SYN, Teardrop, Ping of Death и т.п. Наилучшим способом защиты от обманок IP является верификация адреса отправителя (Source Address Verification), осуществляемая кодом маршрутизации, а не брандмауэром. Посмотрите в своей системе файл /proc/sys/net/ipv4/conf/all/rp_filter. Наличие такого файла говорит о включении механизма верификации адресов Source Address Verification при загрузке системы. Если этого файла нет, внесите приведенные ниже изменения в сценарии инициализации системы до активизации каких-либо сетевых интерфейсов:

(reject) пакеты, идущие с адресов 127.* (зарезервированы для локального интерфейса lo).

Например, в вашей системе используются три интерфейса: eth0, eth1 и ppp0. Мы можем использовать ко-

манду `ifconfig` для определения адреса и маски каждого из этих интерфейсов. Пусть `eth0` использует адрес `192.168.1.0` с маской `255.255.255.0`, `eth1` – адрес `10.0.0.0` с маской `255.0.0.0`, а `ppp0` служит для подклю-

```
# ipchains -A input -i eth0 -s ! 192.168.1.0/255.255.255.0 -j DENY
# ipchains -A input -i ! eth0 -s 192.168.1.0/255.255.255.0 -j DENY
# ipchains -A input -i eth1 -s ! 10.0.0.0/255.0.0.0 -j DENY
# ipchains -A input -i ! eth1 -s 10.0.0.0/255.0.0.0 -j DENY
#
```

Это решение уступает верификации адресов, поскольку при внесении в сеть изменений нужно будет менять правила брандмауэра.

```
# ipchains -A input -i ! lo -s 127.0.0.0/255.0.0.0 -j DENY
#
```

Дополнительные возможности

Существует написанная автором библиотека `userspace`, распространяемая под именем `libfw`. Эта библиотека использует возможность IP Chains версии 1.3 и выше копировать пакеты в пользовательское пространство `userspace` (с помощью конфигурационной опции `IP_FIREWALL_NETLINK`).

Можно задать для пакетов параметры качества обслуживания (QoS) или указать способы рассылки по портам. Я никогда не пользовался этой возможностью, но если вы готовы об этом написать, сообщите мне.

С помощью этой библиотеки в `userspace` можно реализовать такие вещи, как `stateful inspection` (я предпочитаю термин `dynamic firewalling` – динамический брандмауэр).

Другой остроумной идеей является использование контроля для отдельных пользователей с помощью демона `userspace`. Сделать это достаточно просто.

SPF - Stateful Packet Filtering

На сайте <ftp://ftp.interlinx.bc.ca/pub/spf> (проект SPF Брайана Мюррела - Brian Murrell) можно найти информацию по `userspace`. Использование этих возможностей позволяет существенно повысить безопасность систем, подключенных по медленным каналам.

На сайте есть краткая документация, но гораздо важнее поддержка списка рассылки, в котором Брайан отвечает на некоторые вопросы:

- > Я надеюсь, что это именно то, что мне нужно – установка временного «обратного» правила, разрешающего передать пакеты в ответ на исходящий запрос.

Да, это делается именно так. Большинство протоколов понятно, большая часть «обратных» правил верна. Верно, что это поддерживается (это по памяти, простите за ошибки

Проблемы общего плана

Команда `ipchains -L` приводит к «умиранию» системы!

Возможно вы заблокировали работу `DNS lookup` – попробуйте ввести команду с ключом `-n` (использовать адреса вместо имен).

Инверсия не работает!

Вы должны использовать опцию `!` с пробелами с каждой стороны от восклицательного знака. Классический пример ошибки при задании инверсии приведен ниже:

чения к Internet (возможны любые адреса, кроме зарезервированных). В этом случае для решения задачи следует использовать приведенные ниже правила:

Если вы работаете с ядром 2.0, у вас может возникнуть желание защитить и локальный интерфейс `lo` с помощью приведенного ниже правила:

и пропуска) для FTP (активных и пассивных, входящих и исходящих), некоторых приложений `RealAudio`, `traceroute`, `ICMP` и базовых возможностей `ICQ` (прием от серверов `ICQ` и прямые соединения `TCP`, но псевдонимы для вторичных прямых соединений `TCP`, используемых при передаче файлов и в иных случаях, еще не поддерживаются).

> это замена для `ipchains` или просто дополнение?

Это дополнение. Рассматривайте `ipchains` как механизм, позволяющий разрешить или запретить передачу пакетов через Linux-систему. `SPF` представляет собой драйвер, обеспечивающий постоянный мониторинг трафика и сообщающий `ipchains` как следует изменить правила с учетом изменения картины трафика.

ftp-data hack Майкла Азенстайна

Майкл Азенстайн (Michael Hasenstein) из SuSE написал заплатку к ядру, которая добавляет поддержку отслеживания `ftp-соединений` в `ipchains`. Эту заплатку можно найти по ссылке <http://www.suse.de/~mha/patch.ftp-data-2.gz>.

Возможности будущих версий

В версии 2.4 будут переписаны поддержка брандмауэра и NAT. Планы и обсуждение доступны по адресу <http://lists.samba.org>. Эти изменения должны снять множество нерешенных пока вопросов (на деле работа с брандмауэрами и маскарadingом не должна быть столь сложной) и обеспечить более гибкое управление брандмауэрами.

```
# ipchains -A input -i !eth0 -j DENY
#
```

Интерфейса `!eth0` в системе просто не существует, `ipchains` не знает об этом.

Маскарадинг/пересылка не работает!

Убедитесь, что пересылка пакетов (`packet forwarding`) разрешена (в последних ядрах она отключена по умолчанию и пакеты никогда не будут попадать в цепочку `forward`). Вы можете разрешить пересылку пакетов (как пользователь `root`) с помощью команды:

Задачи

Машина с пакетным фильтром:

- PING в любую сеть (особенно полезно это при проверке работоспособности).
- TRACEROUTE для любой сети (очень полезно для диагностики).
- Доступ к DNS

Внутри DMZ:

- Почтовый сервер
 - SMTP во внешние сети
 - SMTP-доступ извне и изнутри
 - Внутренний доступ по протоколу POP-3
- Сервер имен
 - Связь с внешними DNS
 - Доступ к DNS извне, изнутри и с пакетного фильтра.
- Web-сервер
 - HTTP-доступ изнутри и извне
- Rsync-доступ изнутри

Внутренняя сеть:

- Разрешен доступ наружу для WWW, ftp, traceroute, ssh
Это достаточно стандартный набор разрешенных типов доступа. Некоторые предпочитают поначалу открывать для

```
# for f in /proc/sys/net/ipv4/conf/*/rp_filter; do echo 1 > $f; done
#
```

- Установка политики DENY для всех цепочек:
Разрешаем локальный трафик, запрещая все остальное.

```
# ipchains -A input -i ! lo -j DENY
# ipchains -A output -i ! lo -j DENY
# ipchains -A forward -j DENY
#
```

- Установка интерфейсов
Обычно это организуется в сценарии загрузки. Убедитесь, что перечисленные выше операции выполнены до настройки правил для интерфейсов чтобы предотвратить потерю пакетов.

- Вставка модулей маскардинга для каждого протокола.

Нужно установить модули маскардинга для FTP, чтобы из внутренней сети можно было беспрепятственно использовать пассивные и активные сеансы FTP.

```
# insmod ip_masq_ftp
#
```

Фильтрация для транзитных пакетов

При использовании маскардинга лучше всего организовать фильтрацию в цепочке forward.

```
ipchains -A forward -s 192.168.1.0/24 -i eth0 -j good-dmz
ipchains -A forward -s 192.168.1.0/24 -i ppp0 -j good-bad
ipchains -A forward -s 192.84.219.0/24 -i ppp0 -j dmz-bad
ipchains -A forward -s 192.84.219.0/24 -i eth1 -j dmz-good
ipchains -A forward -i eth0 -j bad-dmz
ipchains -A forward -i eth1 -j bad-good
ipchains -A forward -j DENY -l
```

внутренней сети любой доступ наружу и постепенно закрывать лишнее.

- Разрешен доступ к почтовому серверу по протоколу SMTP (обычно передача почты не ограничивается).
- Разрешен доступ к почтовому серверу по протоколу POP-3 (это обеспечивает возможность приема почты).
- Доступ к серверу имен DNS (это нужно для использования WWW, ftp, traceroute и ssh).
- Разрешается rsync для Web-сервера (это позволяет синхронизировать внешний и внутренний web-серверы).
- Открыт доступ к Web-серверу для работы с ним.
- Разрешено использование ping до пакетного фильтра

Это разумная мера – пользователи могут убедиться в работоспособности брандмауэра.

Прежде, чем начать фильтрацию пакетов

- Анти-обманки
Поскольку у нас не используется асимметричная маршрутизация, можно просто включить режим anti-spoofing для всех интерфейсов.

Разделите цепочку forward на несколько пользовательских цепочек по интерфейсам получателей/отправителей – это позволит избавиться от лишних проблем.

```
ipchains -N good-dmz
ipchains -N bad-dmz
ipchains -N good-bad
ipchains -N dmz-good
ipchains -N dmz-bad
ipchains -N bad-good
```

Пакеты со стандартными ошибками ICMP должны восприниматься, поэтому для них создадим специальную цепочку.

```
ipchains -N icmp-acc
```

Установка переходов из цепочки forward

К несчастью (при работе с цепочкой forward) нам известен только выходной интерфейс. Таким образом, для определения входного интерфейса мы будем использовать адрес отправителя (анти-обманки позволяют предотвратить подмену адресов).

Отметим, что мы задаем протоколирование всех операций, которые запрещены правилами (обычно такие события достаточно редки).

тальных случаях управление будет возвращаться в вызвавшую цепочку.

Определение цепочки icmp-acc

Пакеты, содержащие информацию о каких-либо ошибках ICMP, должны восприниматься (ACCEPT), в ос-

```
ipchains -A icmp-acc -p icmp --icmp-type destination-unreachable -j ACCEPT
ipchains -A icmp-acc -p icmp --icmp-type source-quench -j ACCEPT
ipchains -A icmp-acc -p icmp --icmp-type time-exceeded -j ACCEPT
ipchains -A icmp-acc -p icmp --icmp-type parameter-problem -j ACCEPT
```

Доступ изнутри к DMZ (серверы)

Внутренние ограничения:

- Разрешен доступ по WWW, ftp, traceroute, ssh во внешние сети
- Разрешен доступ по SMTP к почтовому серверу
- Разрешен доступ по POP-3 к почтовому серверу
- Разрешен доступ к DNS
- Разрешен доступ по rsync к серверу Web
- Разрешен доступ по WWW к серверу Web

```
ipchains -A good-dmz -p tcp -d 192.84.219.128 smtp -j ACCEPT
ipchains -A good-dmz -p tcp -d 192.84.219.128 pop3 -j ACCEPT
ipchains -A good-dmz -p udp -d 192.84.219.129 domain -j ACCEPT
ipchains -A good-dmz -p tcp -d 192.84.219.129 domain -j ACCEPT
ipchains -A good-dmz -p tcp -d 192.84.218.130 www -j ACCEPT
ipchains -A good-dmz -p tcp -d 192.84.218.130 rsync -j ACCEPT
ipchains -A good-dmz -p icmp -j icmp-acc
ipchains -A good-dmz -j DENY -1
```

Доступ извне к DMZ.

Ограничения DMZ:

- Почтовый сервер
- SMTP для доступа к внешним серверам
- Прием SMTP от внешних и внутренних хостов
- POP-3 от внутренних хостов
- Сервер имен
- Передача DNS во внешние сети

```
ipchains -A bad-dmz -p tcp -d 192.84.219.128 smtp -j ACCEPT
ipchains -A bad-dmz -p udp -d 192.84.219.129 domain -j ACCEPT
ipchains -A bad-dmz -p tcp -d 192.84.219.129 domain -j ACCEPT
ipchains -A bad-dmz -p tcp -d 192.84.218.130 www -j ACCEPT
ipchains -A bad-dmz -p icmp -j icmp-acc
ipchains -A bad-dmz -j DENY
```

Доступ изнутри во внешние сети (Bad).

Внутренние ограничения:

- Разрешен доступ по WWW, ftp, traceroute, ssh во внешние сети
- Разрешен доступ по SMTP к почтовым серверам
- Разрешен доступ по POP-3
- Разрешен доступ к DNS
- Разрешен доступ по rsync к серверам Web
- Разрешен доступ по WWW к серверам Web

```
ipchains -A good-bad -p tcp --dport www -j MASQ
ipchains -A good-bad -p tcp --dport ssh -j MASQ
ipchains -A good-bad -p udp --dport 33434:33500 -j MASQ
ipchains -A good-bad -p tcp --dport ftp -j MASQ
ipchains -A good-bad -p icmp --icmp-type ping -j MASQ
ipchains -A good-bad -j REJECT -1
```

Доступ из DMZ во внутреннюю сеть.

Внутренние ограничения:

- Разрешен доступ по WWW, ftp, traceroute, ssh во внешние сети

- Разрешен ping для брандмауэра

Можно использовать маскардинг для доступа из внутренней сети к DMZ, но мы не будем этого делать. Поскольку никто из внутренней сети не должен пытаться делать недозволённые вещи, мы будем протоколировать все отвергнутые пакеты для идентификации таких попыток.

Отметим, что старые версии Debian используют для pop3 обозначение pop-3 в файле /etc/services, что не соответствует RFC1700.

- Доступ к DNS извне, изнутри и с брандмауэра
- Web-сервер
- Доступ по HTTP извне и изнутри
- Rsync-доступ изнутри
- Другие службы, которые вы можете открыть в DMZ для доступа из внешних сетей.
- Протоколирование не используется, поскольку число записей может быть очень велико.

- Разрешен ping для брандмауэра
- Многие администраторы поначалу разрешают неограниченный доступ во внешние сети и постепенно вводят запреты.
- Протоколирование нарушений.
- Пассивные сеансы FTP обслуживаются модулем masq..
- traceroute использует порты назначения 33434 и выше для UDP.

- Разрешен доступ по SMTP к почтовым серверам
- Разрешен доступ по POP-3
- Разрешен доступ к DNS
- Разрешен доступ по rsync к серверам Web

- Разрешен доступ по WWW к серверам Web
- Разрешен ping для брандмауэра
- Если вы используете маскардинг из внутренней сети в DMZ, просто отвергайте любые па-

```
ipchains -A dmz-good -p tcp ! -y -s 192.84.219.128 smtp -j ACCEPT
ipchains -A dmz-good -p udp -s 192.84.219.129 domain -j ACCEPT
ipchains -A dmz-good -p tcp ! -y -s 192.84.219.129 domain -j ACCEPT
ipchains -A dmz-good -p tcp ! -y -s 192.84.218.130 www -j ACCEPT
ipchains -A dmz-good -p tcp ! -y -s 192.84.218.130 rsync -j ACCEPT
ipchains -A dmz-good -p icmp -j icmp-acc
ipchains -A dmz-good -j DENY -l
```

Доступ из DMZ во внешнюю сеть.

Ограничения DMZ:

- Почтовый сервер
- SMTP для доступа к внешним серверам
- SMTP-доступ извне и изнутри
- Доступ изнутри по POP-3
- Сервер имен

```
ipchains -A dmz-bad -p tcp -s 192.84.219.128 smtp -j ACCEPT
ipchains -A dmz-bad -p udp -s 192.84.219.129 domain -j ACCEPT
ipchains -A dmz-bad -p tcp -s 192.84.219.129 domain -j ACCEPT
ipchains -A dmz-bad -p tcp ! -y -s 192.84.218.130 www -j ACCEPT
ipchains -A dmz-bad -p icmp -j icmp-acc
ipchains -A dmz-bad -j DENY -l
```

Доступ извне во внутреннюю сеть.

Мы не разрешаем никакой (немаскируемый) доступ извне во внутреннюю сеть.

```
ipchains -A bad-good -j REJECT
```

Фильтрация пакетов для брандмауэра

- Если вы хотите использовать фильтрацию для пакетов, адресованных непосредственно
-

```
ipchains -A input -d 192.84.219.1 -j bad-if
ipchains -A input -d 192.84.219.250 -j dmz-if
ipchains -A input -d 192.168.1.250 -j good-if
```

Внешний интерфейс (Bad).

Брандмауэр:

- PING для любой сети
- TRACEROUTE для любой сети

```
ipchains -A bad-if -i ! ppp0 -j DENY -l
ipchains -A bad-if -p TCP --dport 61000:65095 -j ACCEPT
ipchains -A bad-if -p UDP --dport 61000:65095 -j ACCEPT
ipchains -A bad-if -p ICMP --icmp-type pong -j ACCEPT
ipchains -A bad-if -j icmp-acc
ipchains -A bad-if -j DENY
```

Интерфейс DMZ.

Ограничения для брандмауэра:

- PING для любой сети
- TRACEROUTE для любой сети

```
ipchains -A dmz-if -i ! eth0 -j DENY
ipchains -A dmz-if -p TCP ! -y -s 192.84.219.129 53 -j ACCEPT
ipchains -A dmz-if -p UDP -s 192.84.219.129 53 -j ACCEPT
ipchains -A dmz-if -p ICMP --icmp-type pong -j ACCEPT
ipchains -A dmz-if -j icmp-acc
ipchains -A dmz-if -j DENY -l
```

Внутренний интерфейс (Good).

Ограничения для брандмауэра:

- PING для любой сети

кеты, пришедшие иным путем (т. е. Пакеты могут приходить только через корректно организованные соединения).

- Передача DNS во внешние сети
- Восприятие DNS изнутри, извне и от брандмауэра
- Сервер Web
- HTTP-доступ извне и изнутри
- Rsync-доступ изнутри

брандмауэру, потребуется организовать фильтры во входной цепочке. Создадим отдельную цепочку для каждого интерфейса:

```
ipchains -N bad-if
ipchains -N dmz-if
ipchains -N good-if
```

- Организуем переходы в эти цепочки:

- Доступ к DNS
- Внешние интерфейсы также получают отклики на пакеты маскардинга (маскардинг использует порты отправителя 61000 - 65095) и ошибки ICMP для откликов на PING.

- Доступ к DNS
- Интерфейс DMZ также получает отклики DNS, отклики ping и ошибки ICMP.

- TRACEROUTE для любой сети
- Доступ к DNS

Внутренние ограничения:

- Разрешен доступ по WWW, ftp, traceroute, ssh во внешние сети
- Разрешен доступ по SMTP к почтовым серверам
- Разрешен доступ по POP-3
- Разрешен доступ к DNS

```
ipchains -A good-if -i ! eth1 -j DENY
ipchains -A good-if -p ICMP --icmp-type ping -j ACCEPT
ipchains -A good-if -p ICMP --icmp-type pong -j ACCEPT
ipchains -A good-if -j icmp-acc
ipchains -A good-if -j DENY -1
```

- Разрешен доступ по rsync к серверам Web
- Разрешен доступ по WWW к серверам Web
- Разрешен ping для брандмауэра
- Внутренний интерфейс получает запросы и отклики, а также ошибки ICMP.

```
ipchains -D forward 1
ipchains -D output 1
```

Заключение

- Удалите блокирующие правила:
`ipchains -D input 1`

Различия между ipchains и ipfwadm

Часть перечисленных ниже изменений является результатами развития ядра, остальные связаны с отличиями между ipchains и ipfwadm.

1. Изменились многие аргументы – заглавные буквы служат для обозначения команд, строчные – для опций.
2. Поддерживаются произвольные цепочки, поэтому полные имена используются даже для встроенных цепочек (например, input взамен -I).
3. Опция -k больше не поддерживается, используйте ! -u.
4. Опция -b на самом деле вставляет/добавляет/удаляет два правила, а не одно «би-аправленное».
5. Опцию -b можно использовать вместе с командой -C для выполнения 2 проверок (по одной для каждого направления).
6. Опция -x для -l заменена опцией -v.
7. Множественные порты для отправителя и получателя более не поддерживаются. Вместо этого можно использовать диапазоны портов и инверсию.
8. Интерфейсы можно задавать только по именам (не по адресам). Старая семантика была несколько изменена в ядрах серии 2.1.

9. Фрагменты проверяются – автоматическое пропускание отменено.
10. Можно создавать явные цепочки учета трафика.
11. Могут проверяться различные протоколы, работающие поверх IP.
12. Изменен механизм проверки соответствия SYN и АСК (ранее проверка игнорировалась для пакетов, отличных от TCP) – опция SYN не поддерживается больше для правил, не связанных с TCP.
13. Даже на 32-разрядных процессорах поддерживаются 64-разрядные счетчики.
14. Поддерживаются инверсные опции.
15. Поддерживаются коды ICMP.
16. Поддерживаются шаблоны интерфейсов.
17. Манипуляции с TOS проверяются на предмет корректности – старые ядра будут просто отбрасывать пакеты с некорректными опциями TOS; ipchains будет сообщать об ошибке.

Краткая справочная таблица.

[Аргументы в основном указаны ПРОПИСНЫМИ буквами, а опции - строчными]
Отметим, что маскардинг, задаваемый опцией -j MASQ, полностью отличается от -j ACCEPT, и не трактуется лишь как side-effect (как это делается в ipfwadm).

ipfwadm	ipchains	Примечания
-A [both]	-N acct & -I I input -j acct & -I I output -j acct & acct	Создает цепочку acct и передает через нее входные и выходные пакеты
-A in	input	Правило A без указания действия
-A out	output	Правило A без указания действия
-F	forward	Используется как цепочка
-I	input	Используется как цепочка
-O	output	Используется как цепочка
-M -l	-M -l	
-M -s	-M -s	
-a policy	-A [chain] -j POLICY	См. -г и -m
-d policy	-D [chain] -j POLICY	См. -г и -m
-i policy	-I I [chain] -j POLICY	См. -г и -m
-l	-L	
-z	-Z	
-f	-F	
-p	-P	
-c	-C	
-P	-p	
-S	-s	Только для одного порта или диапазона.
-D	-d	Только для одного порта или диапазона.

-V		Используйте -i [name].
-W	-i	
-b	-b	Сейчас реально выполняет 2 правила
-e	-v	
-k	! -y	Не работает без -p tcp
-m	-j MASQ	
-n	-n	
-o	-l	
-r [redirpt]	-j REDIRECT [redirpt]	
-t	-t	
-v	-v	
-x	-x	
-y	-y	Не работает без -p tcp

Примеры трансляции команд ipfwadm

Старая команда	Новая команда
ipfwadm -F -p deny	ipchains -P forward DENY
ipfwadm -F -a m -S 192.168.0.0/24 -D 0.0.0.0/0	ipchains -A forward -j MASQ -s 192.168.0.0/24 -d 0.0.0.0/0
ipfwadm -I -a accept -V 10.1.2.1 -S 10.0.0.0/8 -D 0.0.0.0/0	ipchains -A input -j АСЦЕПТ -i eth0 -s 10.0.0.0/8 -d 0.0.0.0/0

Отметим, что для указания интерфейсов по адресам не существует эквивалента – используйте имена. В приведенном примере интерфейс 10.1.2.1 использует имя eth0.

Использование сценария ipfwadm-wraper

Сценарий ipfwadm-wraper следует использовать вместо ipfwadm для обеспечения обратной совместимости с ipfwadm 2.3a.

Единственным отличием является опция -V, с помощью которой обеспечивается выдача предупреждений. Если используется также опция -W, опция -V будет игнорироваться. В остальных случаях сценарий будет пытаться найти имя интерфейса, связанного с адресом, с помощью команды ifconfig. Если этого не удастся сделать (интерфейс неактивен) сценарий будет заканчивать работу, выдавая сообщение об ошибке.

Выдачу предупреждений можно подавить, задав взамен -V опцию -W или перенаправив стандартный вывод в /dev/null.

Если вы найдете ошибки в сценарии или его реальные отличия от ipfwadm, сообщите мне по адресу rusty@linuxcare.com, указав BUG-REPORT в поле subject (тема). Укажите в этом случае номер старой версии ipfwadm (ipfwadm -h) и версии ipchains (ipchains --version), версии сценария ipfwadm wraper (ipfwadm-wraper --version). Включите также в свое письмо цепочки, выведенные с помощью ipchains-save. Заранее благодарю.

Совместное использование ipchains и ipfwadm-wraper оставляю на ваше усмотрение.

Приложение. Благодарности

Большое спасибо Майклу Нойлингу (Michael Neuling), написавшему первый реализуемый набросок кода цепочек IP, который использовался при работе над документом. Приношу свои извинения за использование идеи кэширования результатов, которую предложил Алан кокс (Alan Cox), а я начал реализацию, увидев ошибки на своем пути.

Спасибо Алану Коксу (Alan Cox) за круглосуточную техническую поддержку по электронной почте и поддержку в работе.

Спасибо всем авторам кода ipfw и ipfwadm, особенно Джосу Восу (Jos Vos). «Находясь на плечах гигантов и все такое...». Это относится к Линусу Торвальдсу (Linus Torvalds) и всем разработчикам программного кода.

Спасибо старательным бета-тестерам и «охотникам за ошибками», особенно Джордану Мендельсону (Jordan Mendelson), Шоу Каррузерсу (Shaw Caruthers), Кевину Моулу (Kevin Moule), д-ру Ливиу Дайя (Dr. Liviu Daia), Хельмуту Адамсу (Helmut Adams), Фрэнку Сикарду (Franck Sicard), Кевину Диттлджону (Kevin Littlejohn), Мэту Кемнеру (Matt Kemner), Джону Хардину (John D. Hardin), Алексею Кузнецову (Alexey Kuznetsov), Леосу Битто (Leos Bitto), Джиму Кунцману (Jim Kunzman), Жерарду Герритсену (Gerard Gerritsen), Сергею Сивкову (Serge Sivkov), Эндрю Бургесу (Andrew Burgess), Стиву Шмидтке (Steve Schmidtke), Ричарду Офферу (Richard Offer), Бернарду Вейсхану (Bernhard Weisshuhn), Ларри Отону (Larry Auton), Амброзу Ли (Ambrose Li), Павлу Краузу (Pavel Krauz), Стиву Чэдси (Steve Chadsey), Франсиско Поротти (Francesco Potorti), Алейну Наффу (Alain Knaff), Касперу Боден-Куммнису (Casper Boden-Cummins) и Генри Холленбергу (Henry Hollenberg).

Переводы

Переводчики документа на другие языки должны помещать свои координаты в начало списка на странице благодарностей, сопровождая их текстом типа: «Special thanks to XXX, for translating everything exactly from my English» (Особая благодарность XXX за точный перевод моего английского). После этого, переводчик должен уведомить об этом автора документа, чтобы последний включил упоминание о переводчике в оригинальный текст.

Arnaud Launay, asl@launay.org,
<http://www.freenix.fr/unix/linux/HOWTO/IPCHAINS-HOWTO.html>

Giovanni Bortolozzo, borto@pluto.linux.it,
<http://www.pluto.linux.it/ldp/HOWTO/IPCHAINS-HOWTO.html>

Herman Rodriguez, herman@maristas.dhis.org,
<http://netfilter.kernelnotes.org/ipchains/spanish/HOWTO.html>