

Маршрут — Интернет

Настроить Linux в качестве маршрутизатора домашней сети не так уж и сложно. Хотя совсем без редактирования конфигурационных файлов не обойтись, большую часть настроек можно проделать с помощью многочисленных графических утилит.

Linux — универсальная операционная система. На основе практически любого дистрибутива можно создать полноценный сетевой сервер. В этой статье мы рассмотрим построение маршрутизатора для небольшой сети на базе компьютера с установленным дистрибутивом Fedora Core 3. Сервер будет осуществлять передачу данных из внутренней сети в Интернет и с помощью NAT транслировать сетевые адреса. Для этого нам потребуется компьютер с процессором не ниже Pentium MMX, 64 Мбайт оперативной памяти, не менее 3 Гбайт дискового пространства, один или два адаптера передачи данных и, конечно, доступ в Интернет. Адаптеры могут быть самыми различными, начиная от стандартных сетевых плат и заканчивая ADSL-модемами с интерфейсом USB. Для того чтобы маршрутизатор заработал, необходимо правильно указать все сетевые реквизиты, то есть прописать IP-адреса или настроить компьютер таким образом, чтобы он получил IP-адрес автоматически.

Поднимаем сетевые интерфейсы

Для настройки сетевых плат выберем в меню пункт «Приложение → Система → Мастер подключения к Интернету» либо «Управление устройствами сети». Подразумевается, что используемые сетевые платы поддерживаются системой.

Выбрав пункт «Настроить», мы попадем в раздел, где отображаются обнаруженные сетевые интерфейсы. Отметив нужный и нажав на кнопку «Изменить», перейдем непосредственно в меню настройки сетевых плат. Здесь задается IP-адрес, маска сети и адрес шлюза. То есть, настраивается статический маршрут, если подключение к другим сетям осуществляется через сетевую плату. Если же доступ происходит через модем, сетевые реквизиты мы получим автоматически при подключении к удаленному хосту.

Конфигурационный файл для каждого сетевого устройства находится в каталоге `/etc/sysconfig/network-scripts` и называется `ifcfg-eth0` — для первого интерфейса, `ifcfg-eth1` — для второго и т. д. Если вы хотите модифицировать сетевые адреса вручную или добавить еще одно устройство на новом интерфейсе, потребуется изменить некоторые строки в этих файлах:

DEVICE = имя физического сетевого устройства

IPADDR = IP-адрес

NETMASK = маска сети

NETWORK = IP-адрес подсети

BROADCAST = широковещательный IP-адрес

ONBOOT = будет ли интерфейс активным во время загрузки

BOOTPROTO = переменная может принимать значения:

«none» — не использовать протокол во время загрузки

«bootp» — использовать протокол BOOTP

«dhcp» — использовать протокол DHCP

USERCTL = может принимать одно из значений:

«yes» — любой пользователь может контролировать устройство

«no» — только root может контролировать устройство

Базовые конфигурационные файлы сети: `/etc/hostname`, в котором хранится доменное имя вашего компьютера, например `my.domain.ru`, а также `/etc/resolv.conf` — файл, используемый резолвером для определения IP-адреса ресурса по его имени. Данный файл, как правило, выглядит следующим образом:

order bind,hosts — поиск имени сперва через DNS, а затем в файле `/etc/hosts`

«multi on» — у нас машины с несколькими сетевыми адресами

«nospoof on» — проверять подмену IP-адресов (IP spoofing)

Теперь рассмотрим строение файла `/etc/hostname`:

search domain.ru

«nameserver» — любой IP-адрес, на котором настроен DNS и к которому есть доступ

Опция «order» применяется для определения порядка использования сервисов. В примере установлено, что вначале резолвер обращается к DNS-серверу, а затем к файлу `/etc/hosts`. Опция «multi» говорит, что компьютеры, описанные в файле `/etc/hosts`, могут иметь не один адрес, а несколько (соответственно, и несколько сетевых интерфейсов). Например, шлюз

всегда имеет несколько адресов, и у него эта опция должна быть всегда установлена в положение «ON». Опция «nospoof» предписывает не разрешать подмену адресов. «IP spoofing» — это способ атаки, при котором удаленный компьютер представляется тем, чем он не является на самом деле. Теперь включим возможность пересылки пакетов. Для этого отредактируем файл `/etc/sysconfig/network`:

```
FORWARD_IPV4 = yes
```

А также `/etc/sysctl.conf`:

```
net.ipv4.ip_forward = 1
```

Чтобы изменения вступили в силу, необходимо перезагрузить сетевой сервис следующим образом:

```
/etc/rc.d/init.d/network restart
```

Просмотрим информацию о сетевых интерфейсах:

```
ifconfig
```

```
lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:1667 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1667 errors:0 dropped:0 overruns:0 carrier:0
```

Настройка модемного соединения

Для подключения модема воспользуемся «Мастером подключения к Интернету». Выберем подключение обычного модема. Система попытается самостоятельно определить устройство, после чего необходимо будет задать основные настройки: номер телефона (если вы звоните через офисную АТС, то перед номером не забудьте указать код выхода на городскую линию), по которому

мы будем звонить, название учетной записи, логин и пароль для входа в Интернет или удаленную сеть. После ввода данных в систему будет добавлено устройство, обозначенное как «Modem».

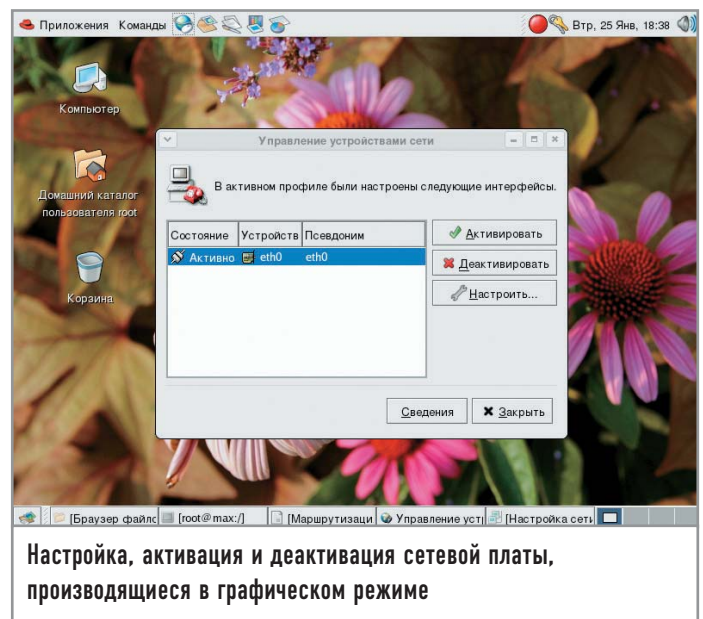
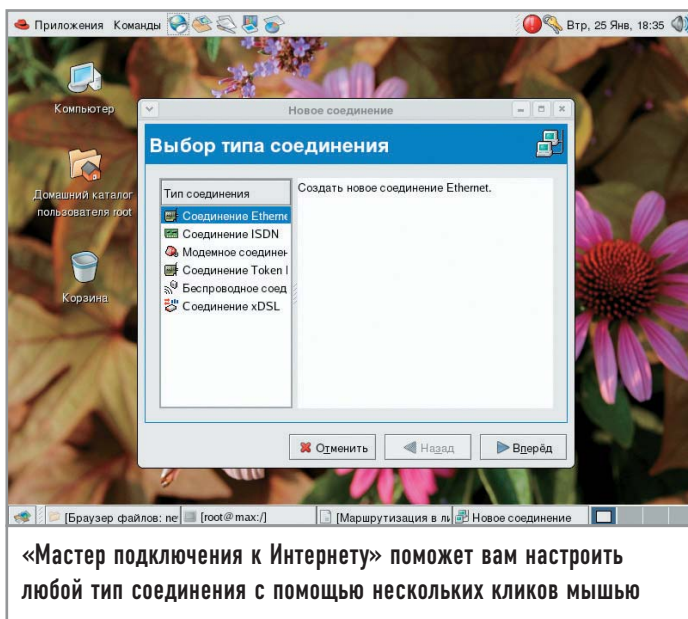
В этом же меню выберем вкладку «Оборудование». Здесь можно произвести настройки модема: установить порт (для COM1 в Unix-системах принято обозначение `ttyS0`), выбрать тональный или пульсовый набор номера, определить скорость порта (по умолчанию 115 200 кбит/с), громкость динамика и тип управления потоком данных (по умолчанию — аппаратный). Все данные настройки модема можно изменить, зайдя в меню «Приложения → Система → Управление устройствами сети». Выберите пункт «Изменить». На закладках этого диалогового окна можно настроить необходимые параметры модема. Укажите адрес сервера DNS и, если это необходимо, задайте для модема установку соединения сразу при включении компьютера. На вкладке «Маршрут» можно добавить статический маршрут. Раздел меню «Провайдер» дает возможность изменить данные учетной записи (логин, пароль, номер телефона). В пункте «Дополнительно» настраиваются свойства программы для дозвона.

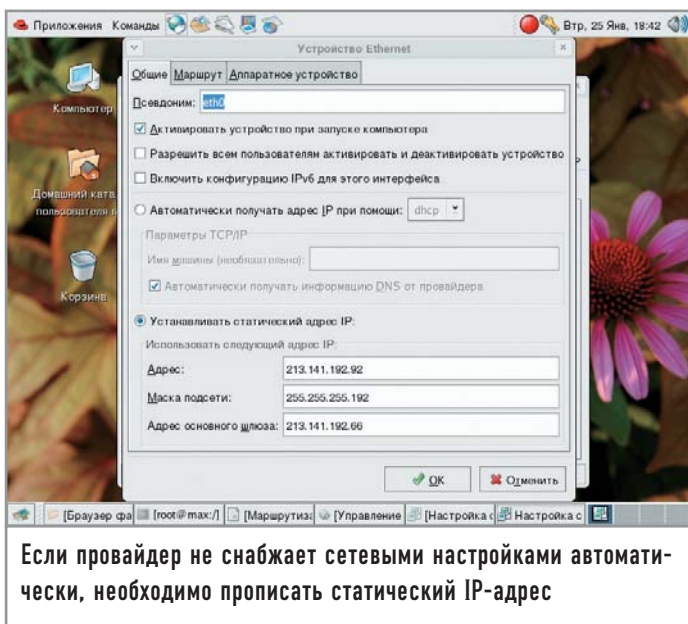
Первый звонок

Дозвон можно осуществить несколькими способами. Самый простой путь — сделать это через графический интерфейс. В меню «Настройка сети» нужно выделить необходимое устройство и нажать кнопку «Активировать». Если модем настроен правильно, то он сразу станет звонить по указанному номеру. Утилита, отвечающая за дозвон, называется `Wvdial`. Ее конфигурационный файл — `/etc/wvdial.conf`. В нем уже записаны все настройки, которые мы проделали через вышеописанные разделы меню.

Второй вариант — использовать для дозвона демон `pppd`. Для этого потребуется немного отредактировать файл `/etc/ppp/options`, поместив в него следующие строки:

```
modem
crtsets
asynsmap 0
```





Команда для запуска rppd:

```
rppd ttyS0 57600 lock connect 'chat -V -f /etc/ppp/ISP1'
defaultroute noipdefault debug nodetach
```

Теперь мы создадим скрипт, в котором будут прописаны основные настройки с данными для программы chat (номер телефона, логин, пароль и т. д.), и назовем его /etc/ppp/ISP1:

ABORT "ERROR" — напротив **ABORT** строка, получив которую, chat завершит работу

ABORT "NO DIALTONE"

TIMEOUT 5

«» «AT» — символы «» означают, что от модема ничего ждать не стоит и нужно сразу посылать ему команду AT

«OK» «ATZ» — сбрасываем настройки модема

«OK» «ATS7 = 120» — указываем модему, сколько времени ждать соединения

ABORT «BUSY»

ABORT «NO ANSWER»

ABORT «NO CARRIER»

ABORT «Login Incorrect»

«OK» «ATDP(9)1234567» — номер телефона, по которому осуществляется дозвон

TIMEOUT 125

«CONNECT» «\c» — после получения строки «CONNECT» мы уменьшаем время ожидания до 30 секунд; строка «\c» говорит, что в модем посылать ничего не нужно и можно сразу же приступить к проверке логина и пароля

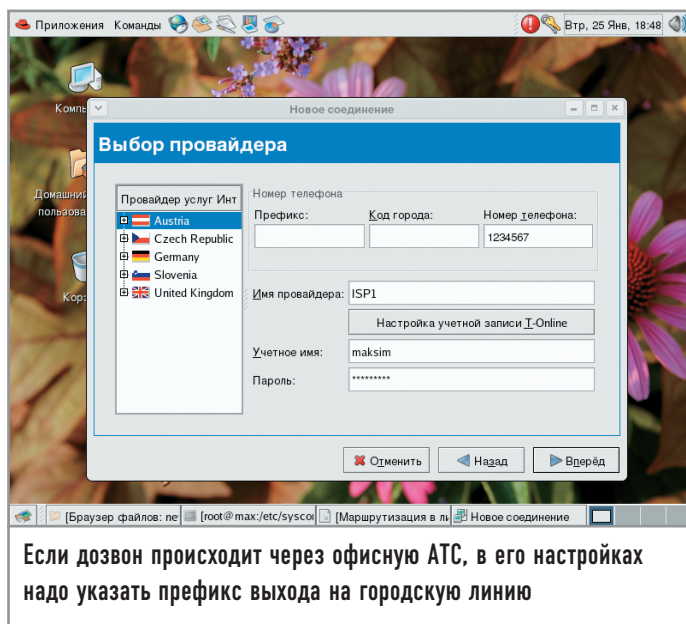
TIMEOUT 30

«login» «maksim»

«password» «1234567»

«}» «\c» — убираем с экрана ненужные символы

После дозвона мы можем проверить соединение, набрав команду ifconfig и посмотрев, есть ли в списке устройство «rppd0», или же просто отдав команду «ping» на любой хост в Интернет-



те. На этом настройку доступа в Интернет через модем можно считать завершенной, однако давайте рассмотрим еще несколько возможностей подключения к сети.

Настройка пакетного фильтра

Итак, тем или иным способом мы подключились к сети, маршрутизация включена, теперь можно приступать к работе. Но если вам предоставлен только один реальный IP-адрес для интерфейса, через который вы соединились с Интернетом, то о какой маршрутизации может идти речь? Решить эту проблему можно либо при помощи сервиса NAT, либо обычного прокси-сервера. Далее мы рассмотрим настройку с помощью обоих этих способов.

На данный момент Fedora Core обладает мощным средством обработки сетевых данных — пакетным фильтром iptables. Номер версии программы на нашем компьютере был 1.2.11. Данный фильтр включен в ядро системы, поэтому перекомпилировать его не придется. С помощью iptables можно настроить трансляцию адресов, обеспечить безопасность сервера со стороны Интернета, а также ограничить доступ к машине из внутренней сети. Проверить версию и работу фильтра можно командой:

iptables -V

Сетевой пакет проходит через любой интерфейс (даже через работающий по умолчанию loopback), ядро проверяет этот пакет по определенным правилам: «ACCEPT» — позволяет пройти пакету, «DROP» — не позволяет, «REJECT» — пакет отбрасывается. Кроме того, он может быть отправлен через существующие цепочки правил. Для управления этими самыми правилами используется одноименная программа iptables, которая устанавливается на всех системах по умолчанию. Все правила обработки пакетов хранятся в памяти до перезагрузки системы. Чтобы сохранить выполненные самостоятельно настройки «iptables», необходимо записать их в файл, которым изначально является /etc/rc.d/rc.firewall. Ак-

тивация данного скрипта может выполняться как вручную, так и добавляться в автозапуски системы. Не забудьте дать файлу права на выполнение:

```
chmod 744 /etc/rc.d/rc.firewall
```

Обновленную таблицу iptables можно просмотреть в окне терминала с помощью следующей команды:

```
iptables -L
```

Существуют три основные цепочки правил. «INPUT» — для входящих пакетов, «OUTPUT» — для исходящих, «FORWARD» — для пакетов, проходящих через данную машину к другой. Если пакет не удовлетворяет условиям цепочки («ACCEPT»), он попадает в следующую («REJECT»), а потом еще в одну («DROP»). Если же пакет не попал ни под одно из вышеописанных правил в определенной цепочке, он отбрасывается или же останавливается в любом случае. Для того чтобы просмотреть существующие на данный момент цепочки, введем необходимую команду:

```
iptables -P
```

Для создания новой цепочки понадобится команда:

```
iptables -N
```

Для добавления правила в цепочку введем строку:

```
iptables -A -j
```

Также можно задать следующие условия:

```
-p
--dport
```

```
--sport
-s
-d
-i — только для INPUT
--icmp-type — для icmp
```

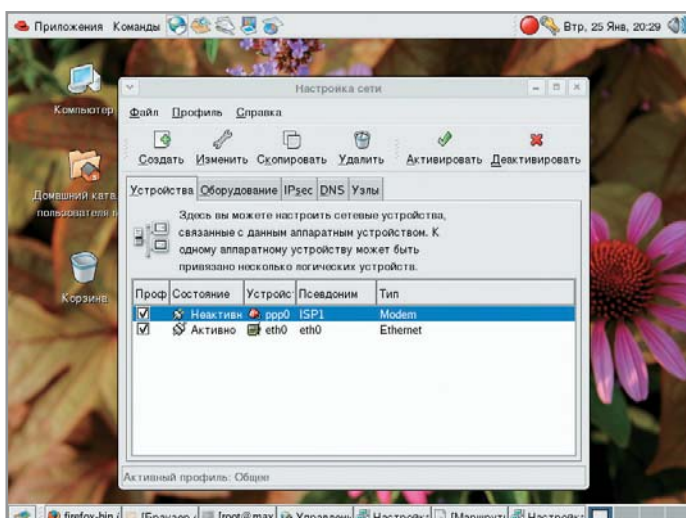
Составление правил

Теперь мы можем создать собственные правила, удовлетворяющие следующим условиям:

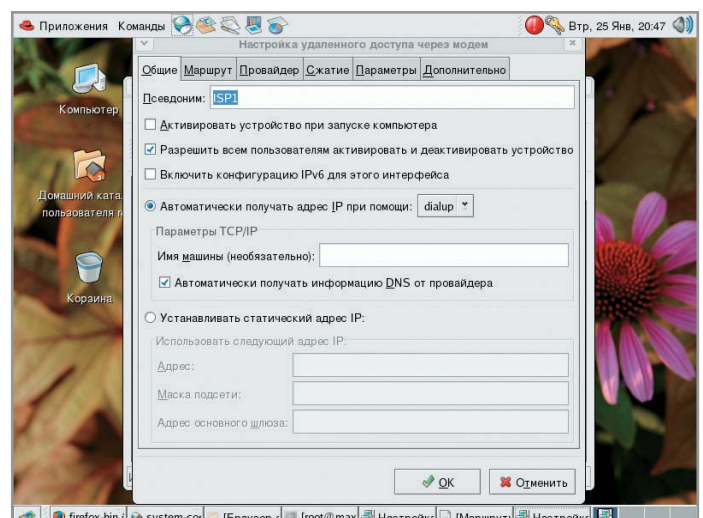
- ▶ Пакеты локального интерфейса будут выходить в сеть без ограничений.
- ▶ Большинство входящих соединений (на основную часть портов) будут закрыты.
- ▶ Порты, необходимые для работы FTP (21) и DNS (53).

Создадим файл rc.firewall и отредактируем его:

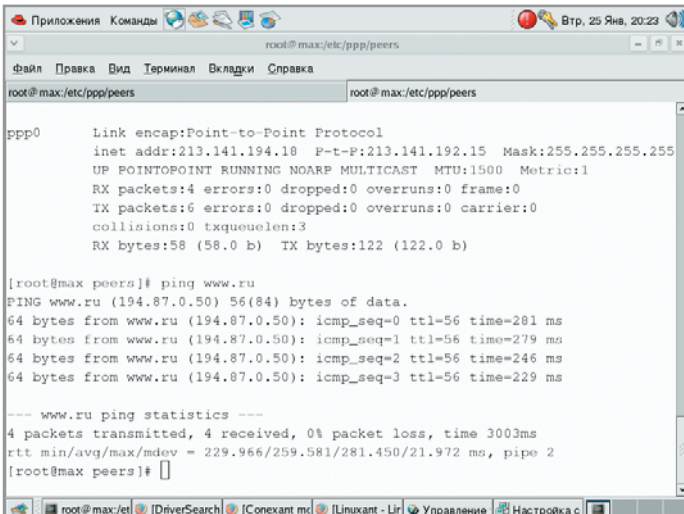
```
#!/bin/sh
# Опишем интерфейс выхода в сеть (если модем — то ppp0;
# если сеть — то eth0)
INET_IFACE = "eth0"
# На всякий случай укажем полный путь к iptables
IPTABLES = "/sbin/iptables"
# Включаем действия по умолчанию для стандартных цепочек
# правил
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -P FORWARD DROP
# Удаляем все правила, если они есть
$IPTABLES -F
$IPTABLES -X
# Создаем наши правила, для того чтобы обезопасить
# работу с сетью
# Отбрасываем TCP с неправильными флагами
$IPTABLES -N bad_tcp_packets
# TCP, прошедшие основную проверку
$IPTABLES -N allowed
```



Дозваниваться можно с помощью встроенных средств GNOME, просто активировав устройство «Modem»



В расширенном диалоге настроек модема можно выбрать метод сжатия, ввести инициализационную строку и реквизиты провайдера



```

root@max:/etc/ppp/peers# cat /etc/ppp/peers/ppp0
Link encap:Point-to-Point Protocol
inet addr:213.141.194.18 P-t-P:213.141.192.15 Mask:255.255.255.255
UP POINTOPOINT RUNNING NOARP MULTICAST MTU:1500 Metric:1
RX packets:4 errors:0 dropped:0 overruns:0 frame:0
TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:3
RX bytes:58 (58.0 b) TX bytes:122 (122.0 b)

[root@max peers]# ping www.ru
PING www.ru (194.87.0.50) 56(84) bytes of data:
64 bytes from www.ru (194.87.0.50): icmp_seq=0 ttl=56 time=281 ms
64 bytes from www.ru (194.87.0.50): icmp_seq=1 ttl=56 time=279 ms
64 bytes from www.ru (194.87.0.50): icmp_seq=2 ttl=56 time=246 ms
64 bytes from www.ru (194.87.0.50): icmp_seq=3 ttl=56 time=229 ms

--- www.ru ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 229.966/259.581/281.450/21.972 ms, pipe 2
[root@max peers]#

```

Командная строка — удобное средство просмотра параметров сетевых интерфейсов и тестирования соединения

Пакеты основных протоколов

```
SIPTABLES -N tcp_packets
```

```
SIPTABLES -N udp_packets
```

```
SIPTABLES -N icmp_packets
```

Разрешаем прохождение пакетов для локального интерфейса

```
SIPTABLES -A INPUT -i lo -j ACCEPT
```

```
SIPTABLES -A OUTPUT -o lo -j ACCEPT
```

Защищаемся от DOS- и SYN-атак

```
SIPTABLES -A bad_tcp_packets -p tcp --tcp-flags SYN,ACK
```

```
SYN,ACK -m state --state NEW -j DROP
```

Пропускаем пакеты уже установленных соединений

```
SIPTABLES -A allowed -p TCP --syn -j ACCEPT
```

```
SIPTABLES -A allowed -p TCP -m state --state ESTAB-
```

```
LISHED,RELATED -j ACCEPT
```

А все остальные из этой цепочки сбрасываем

```
SIPTABLES -A allowed -j DROP
```

Здесь можно открыть некоторые порты, так как по умолчанию мы закрыли все

Открываем 21-й порт для FTP

```
SIPTABLES -A tcp_packets -p TCP -s 0/0 --dport 21 -j allowed
```

Открываем 53-й порт для DNS

```
SIPTABLES -A udp_packets -p UDP -s 0/0 --sport 53 -j
```

```
ACCEPT
```

Отбрасываем пакеты ICMP

```
SIPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 3 -j
```

```
ACCEPT SIPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-
```

```
type 11 -j ACCEPT SIPTABLES -A icmp_packets -p ICMP -s
```

```
0/0 --icmp-type 12 -j ACCEPT
```

Далее для нашего же удобства проводим проверку по всем протоколам отдельно

```
SIPTABLES -A INPUT -p ALL -i $INET_IFACE -m state --
```

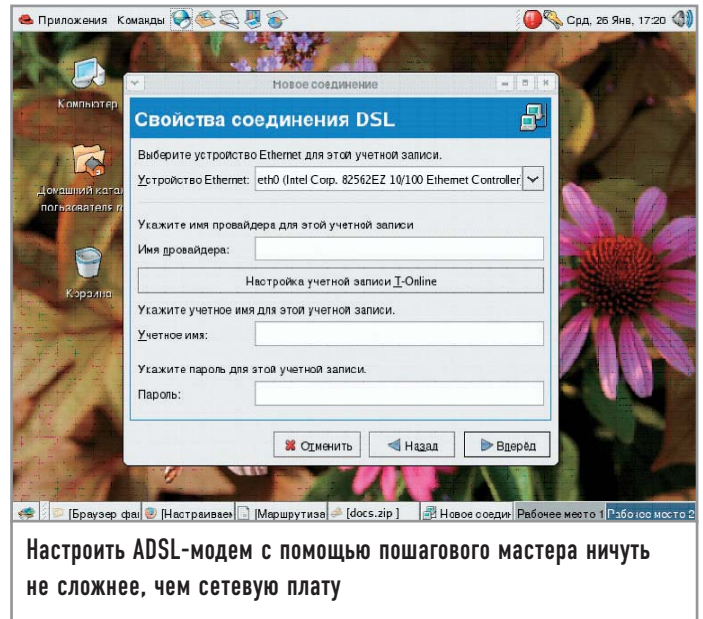
```
state ESTABLISHED,RELATED -j ACCEPT
```

```
SIPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
```

```
SIPTABLES -A INPUT -p UDP -i $INET_IFACE -j udp_packets
```

```
SIPTABLES -A INPUT -p ICMP -i $INET_IFACE -j
```

```
icmp_packets
```



NAT и masquerade

Маскарад (masquerade) и трансляция адресов (NAT) в мире Linux не являются синонимами. Маскарад — замена адреса нашей машины на адрес сервера, выполняющего маскарадинг. Трансляция адресов — замена адреса на любой указанный. Как же работает маскарад? Наш пакет (например, на www.ru) проходит через сервер, и в нем адрес источника меняется на адрес сервера (123.123.123.123). Пакет приходит на www.ru, и хост отвечает по адресу пакета (123.123.123.123). Так как наш сервер запомнил, что пакет для www.ru посылали мы, то он принимает пакет и отдает его нашему компьютеру. Вот и все, пакет ушел и вернулся. Включим маскарад в iptables, это очень просто:

```
iptables -A FORWARD -s 192.168.1.0/24 -j ACCEPT
```

Данной командой мы разрешили прохождение пакетов между сетевыми интерфейсами из локальной сети 192.168.1.0/24.

```
iptables -A FORWARD -d 192.168.1.0/24 -j ACCEPT
```

Эта команда разрешает прохождение пакетов между сетевыми интерфейсами в локальную сеть 192.168.1.0/24. Таким образом, с помощью последней включим маскарад для сети 192.168.1.0/24.

```
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.1.0/24 -j MASQUERADE
```

Теперь необходимо проверить, включена ли маршрутизация в ядре?

```
cat /proc/sys/net/ipv4/ip_forward
```

Если на выходе получено значение «1», все в порядке. Если «0» — нужно включить маршрутизацию следующей командой:

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

NAT работает так: наш пакет, отправленный на www.ru, проходит через сервер, и в нем адрес источника меняется на указанный. Пакет приходит на www.ru, и хост отвечает по адресу в пакете (123.123.123.123). Затем пакет приходит на наш сервер, и происходит обратная замена. NAT включается в iptables вот так:

```
iptables -A FORWARD -s 192.168.1.0/24 -j ACCEPT
iptables -A FORWARD -d 192.168.1.0/24 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.1.0/24
-j SNAT --to-source 123.123.123.123
```

Для облегчения работы с iptables написано несколько графических приложений, которые при наличии среды Xwindow помогают управлять пакетным фильтром. Firewall Builder позволяет управлять цепочками правил для различных пакетных фильтров, в том числе и для iptables. Найти ее можно здесь: fwbuilder.sf.net.

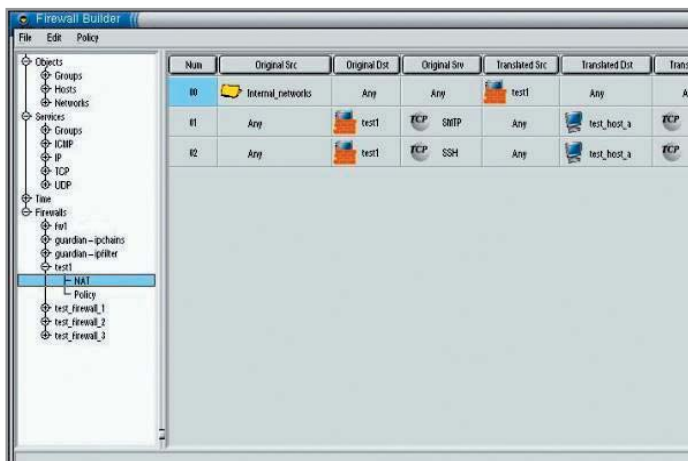
Настройка Squid

Вполне возможна ситуация, когда через роутер вам потребуется получить доступ только к веб-страницам. Причем доступ нужно организовывать на основе пользовательских прав, вплоть до того, чтобы раздавать ограничения пользователям одного домена или определять доступность тех или иных сайтов. А если необходимо получать статистику посещений веб-ресурсов? В таком случае для этих и многих других целей прекрасно подойдет прокси-сервер, например Squid. Некоторые функции программы необходимо обозначить на этапе компиляции. Скачаем исходные тексты последней стабильной версии Squid с официального сайта разработчиков www.squid-cache.org и приступим к установке:

```
tar -zxvf squid-2.5.STABLE5.tar.gz
cd ./squid-2.5.STABLE5
./configure
```

При конфигурации полезно указать следующие опции:

```
--prefix = PREFIX — прописать нужный путь для установки
--enable-delay-pools — ограничить полосы пропускания
```



Те, кто предпочитает графическую оболочку, могут настраивать правила для firewall с помощью программы Firewall Builder

```
--enable-useragent-log — включить запись сведений о браузере
пользователя в лог
```

```
--enable-err-language = lang — включить функцию вывода
ошибок на родном языке
```

Далее компилируем и устанавливаем программу:

```
make && make install
```

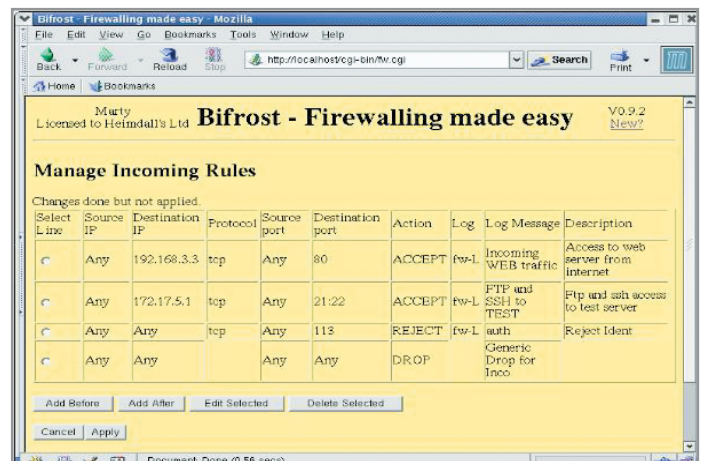
Теперь приступаем к непосредственной настройке сервера, отредактировав файл /usr/local/etc/squid.conf:

```
http_port 3128 — задаем порт (по умолчанию 3128).
cache_mem 20 MB — сколько оперативной памяти Squid
может забрать под свои нужды
cache_dir /usr/local/squid/cache 500 16 256 — указывает
Squid, где сохранять кешируемые файлы, указывает отдать
под кеш 500 Мбайт и создать 16 и 256 каталогов 1-го и 2-го
уровня соответственно. Не забудем также выставить права
пользователя, под именем которого у нас будет работать Squid
cache_access_log /usr/local/squid/logs/access.log
cache_log /usr/local/squid/logs/cache.log
cache_store_log /usr/local/squid/logs/store.log
acl net src 192.168.1.0/255.255.255.0
acl all src 0.0.0.0/0.0.0.0
acl net src 192.168.1.0/255.255.255.0
http_access allow net
http_access deny all
```

Запускать Squid первый раз надо с параметром создания кеша, используя ключ -z:

```
/usr/local/squid/bin/squid -z
```

Теперь, после всех проделанных манипуляций, выход в Интернет в нашей сети происходит через прозрачный прокси-сервер. Самое главное — не забыть указать Linux запускать Squid при каждом перезапуске системы. |



С помощью имеющей веб-интерфейс программы Bifrost настраивать брандмауэр можно даже удаленно