Сетевые средства Linux

2-е издание, исправленное

Бражук А.И.

Национальный Открытый Университет "ИНТУИТ"
2016

Сетевые средства Linux/ А.И. Бражук - М.: Национальный Открытый Университет "ИНТУИТ", 2016

Рассматриваются вопросы настройки сетевых интерфейсов, маршрутизации, фильтрации трафика базовыми средствами Linux, технологии трансляции сетевых адресов (NAT), а также диагностики сетевых подключении приложениями протокола ICMP и средствами анализа сетевого трафика.

В рамках курса рассматриваются вопросы использования программы VirtualBox для изучения сетевых возможностей Linux, настройки сетевых интерфейсов и маршрутизации сетевого трафика с помощью пакетов iproute2, команд ifconfig, arp, route, конфигурационных файлов сетевой подсистемы, особенности транспортного и прикладного уровней, диагностики сетевых подключений с помощью соответствующих утилит (ping, traceroute, netstat) и анализа сетевого трафика средствами tcpdump, организации на основе Linux маршрутизатора, межсетевого экрана и системы доступа в Интернет на сетевом уровне, использования технологий виртуальных частных сетей. Изложение материала сопровождается примерами, типичных для условий небольшой сети. Приведенные примеры можно легко воспроизвести в среде VirtualBox.

Использование виртуализации для изучения Linux

Приведены элементарные сведения о Linux, работе со средой виртуализации VirtualBox и сетевых возможностях VirtualBox. В качестве упражнения предложено создать виртуальный Linux-компьютер.

Содержание:

Особенности использования Linux

В широком смысле под термином Linux понимают группу Unix-подобных операционных систем, основанных на открытом ядре Linux (нач. 90-х годов — Линус Торвальдс, университет Хельсинки) и различных библиотеках, утилитах и программах. Linux тесно связан с понятиями свободного программного обеспечения (free) и программного обеспечения с открытым исходным кодом (open-source). Данные понятия подкреплены юридически соотвествующими лицензиями для ПО (GPL, BSD и т.д); обычно обозначают ПО с доступным исходным кодом (бинарный код запрещено распространять без исходного) и возможностью свободного использования, модификации и распространения.

В узком смысле под термином Linux имеют в виду данный дистрибутив Linux — вариант работоспособной Linux-системы, включающий компиляцию ядра, набор системных скриптов и конфигурационных файлов, а также различное программное обеспечение. Условно можно сказать, что дистрибутив состоит из следующих компонентов: набора пакетов (форма представления ПО в дистрибутиве, включающая бинарный код, настройки, помощь и документацию или исходный код с настройками компиляции какого-либо вида ПО); ПО для управления, обновления и распространения пакетов (управление репозиториями ПО, загрузка ПО из удаленных репозиториев, определение и устранение зависимостей, средства создания пакетов и т.д.); ПО для инсталляции дистрибутива.

Существует множество дистрибутивов Linux, различающихся основным форматом пакетов, функциональным назначением, набором программного обеспечения, поддержкой различных аппаратных платформ и т.д. В данном пособии используется Debian GNU/Linux или просто Debian, характеризующийся, в частности, развитой системой управления программным обеспечением и большой базой программного обеспечения, содержащегося в стандартных и дополнительных репозиториях.

Основными способами взаимодействия и управления системой, используемыми в данном пособии, являются интерфейс командной строки и редактирование конфигурационных файлов. Интерфейс командной строки — такой способ взаимодействия с системой, при котором каждая строка, передаваемая пользователем системе, интерпретируется системой как команда, которую необходимо выполнить. Команды также можно группировать в пакетные исполняемые файлы (сценарии) для решения системных и прикладных задач. Концепция конфигурационных файлов подразумевает использование для хранения настроек системы и прикладного ПО текстовых файлов, формат которых одновременно воспринимаем пользователем и программным обеспечением. Необходимые знания по использованию интерфейса командной строки можно получить из соответствующих учебных пособий [1][3] или справочника к дистрибутиву [4].

Возможны различные варианты установки Linux на компьютер:

- Монопольно одну ОС (в данном случае можно использовать автоматическое разбиение диска программой установки Linux).
- В качестве альтернативной ОС. Выбор загружаемой ОС происходит в меню при включении компьютера. При таком способе установки обязательно необходимо сделать резервные копии ценных данных; высвободить место на жестком диске для разделов Linux (минимально требуются: раздел для корневой файловой системы и раздел подкачки). Далее в процессе установки потребуется установить в загрузочном секторе диска (MBR – Master Boot Record) загрузчик-меню выбора ОС (стандартный Linux – grub или lilo, или альтернативный).
- Использовать LiveCD – ОС, загружаемая с CD или DVD.

- Как виртуальную машину.

В рамках данного курса рекомендуется использовать последний вариант. Среда виртуализации не требует менять конфигурацию дисковых разделов: диски виртуальной машины будут представлены в виде файлов на диске компьютера; позволяет легко переключаться между основной системой и виртуальными машинами. Виртуальная сеть дает возможность отлаживать различные сетевые конфигурации, запуская на одном физическом компьютере две и более систем.

Программное обеспечение виртуализации VirtualBox

Виртуализация (в данном контексте) - возможность запускать несколько виртуальных операционных систем (гостевых операционных систем) одновременно на одном физическом компьютере (хосте), то есть фактически создать несколько виртуальных компьютеров. Данная возможность реализуется средствами специального ПО — гипервизора или менеджера виртуальных машин.

В данной работе в качестве гипервизора используется программное обеспечение Oracle/SUN VirtualBox ссылка: http://virtualbox.org. VirtualBox должен быть установлен как приложение операционной системы физического компьютера (хостовой системы). В качестве хостовых систем поддерживаются различные версии Windows, Linux, Solaris и MAC OS X (соответствующий пакет можно скачать с сайта).

Управление виртуальными машинами осуществляется с помощью средства графического интерфейса пользователя, которое называется менеджер VirtualBox (VirtualBox manager). Рабочее окно программы (рис. 1.1.) содержит текстовое меню, набор кнопок, слева - область со списком виртуальных машин, справа — вкладки "Детали", "Снимки", "Описания", относящиеся к выбранной в списке виртуальной машине.

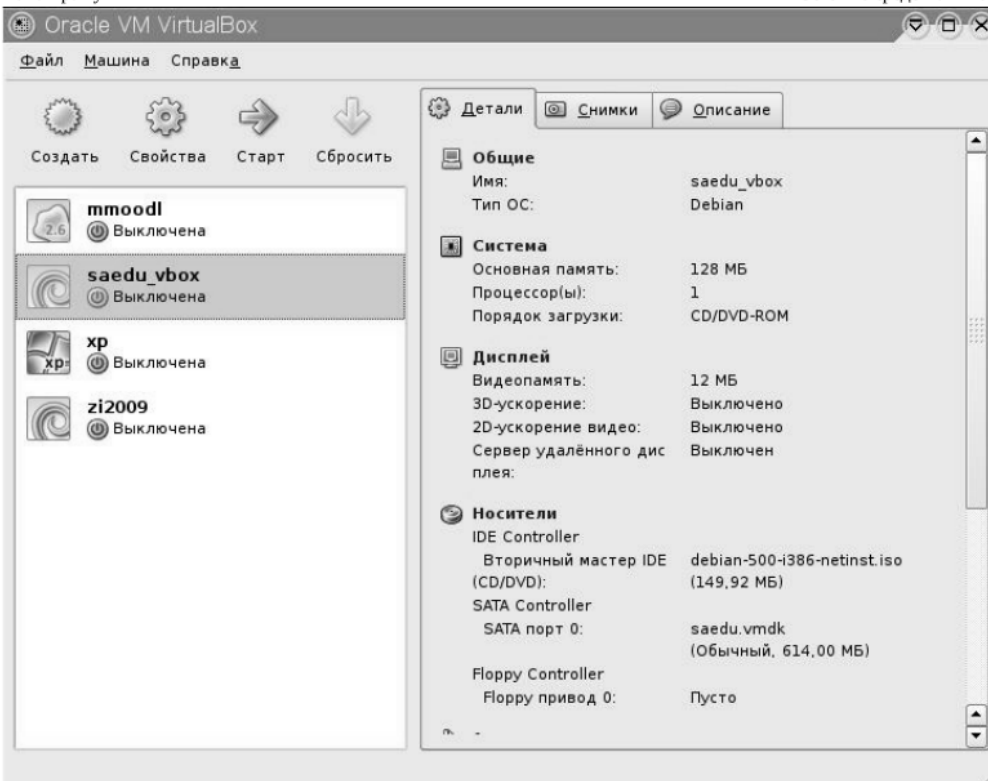

Рис. 1.1. Менеджер VirtualBox.

Кнопка " Создать " предназначена для вызова мастера создания виртуальной машины. В процессе работы мастера пользователю необходимо указать имя виртуальной машины, выбрать тип и версию операционной системы, указать размер памяти для виртуальной машины, создать новый виртуальный жесткий диск или использовать файл существующего жесткого диска.

Кнопка " Свойства " открывает окно настроек данной виртуальной машины. В частности, в списке свойств пункт "Носители" позволяет подключить к виртуальному приводу оптических дисков диск, вставленный в привод физического диска, а также iso-образ CD/DVD-диска (рис. 1.2.)

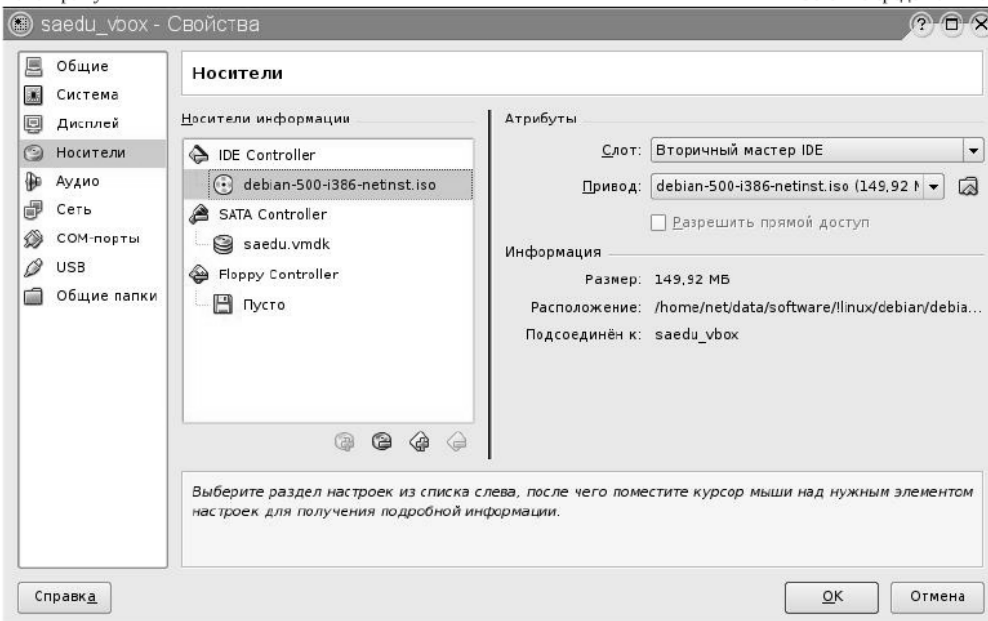

Рис. 1.2. Подключение образа CD-диска к оптическому приводу виртуальной машины.

Кнопка " Старт " предназначена для запуска виртуальной машины. Виртуальная машина запускается в новом окне (рис 1.3.). Если кликнуть мышью в какую-либо область окна ВМ, то клавиатура и мышь перехватываются ВМ, для возврата управления в основную систему следует нажать хостовую клавишу (по умолчанию правый Ctrl).

Рис. 1.3. Работающая ВМ в среде VirtualBox.

Меню окна работающей ВМ позволяет: включить полноэкранный режим, послать сигнал Ctrl+Alt+Del виртуальной машине, сделать паузу в работе ВМ, выключить ВМ, а также управлять различными устройствами, подключаемыми к ВМ.

Указать порядок выбора носителей для загрузки можно в свойствах виртуальной машины в пункте "Система" на вкладке "Материнская плата" (рис. 1.4).

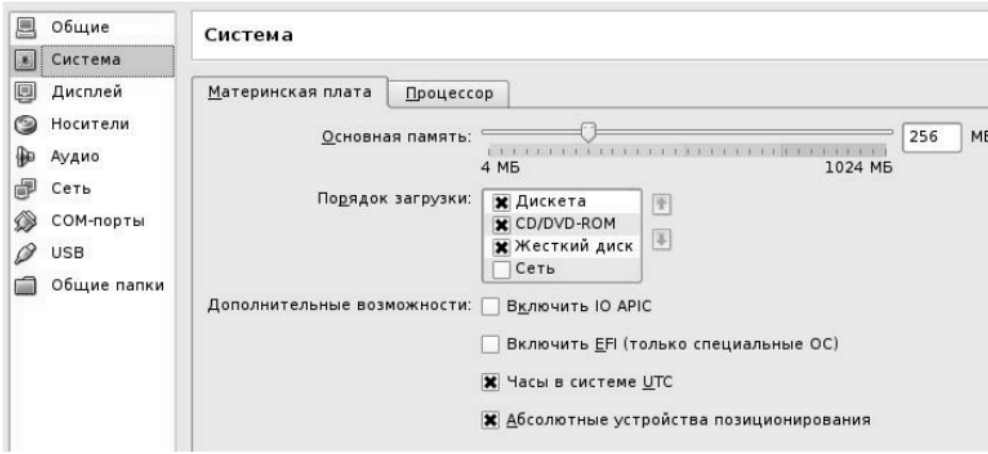

Рис. 1.4. Вкладка "Материнская плата" пункта "Система" свойств ВМ

Перенос виртуальных машин между хостовыми системами осуществляется посредством импорта/экспорта: на источнике нужно сделать импорт ВМ (Файл — Импорт конфигурации), перенести полученные файлы на систему-назначение и на ней выполнить экспорт ВМ (Файл — Экспорт конфигурации).

Основные приемы и особенности работы с VirtualBox описаны в руководстве пользователя ссылка: http://www.virtualbox.org/manual/UserManual.html .

Сетевые возможности VirtualBox

Гипервизор VirtualBox позволяет виртуальным машинам использовать сетевые подключения хостовой системы, а также создавать виртуальные сети для виртуальных машин. С помощью средств графического интерфейса пользователя для ВМ может быть настроено до 4-х сетевых адаптеров (пункт "Сеть" свойств виртуальной машины - рис. 1.5).

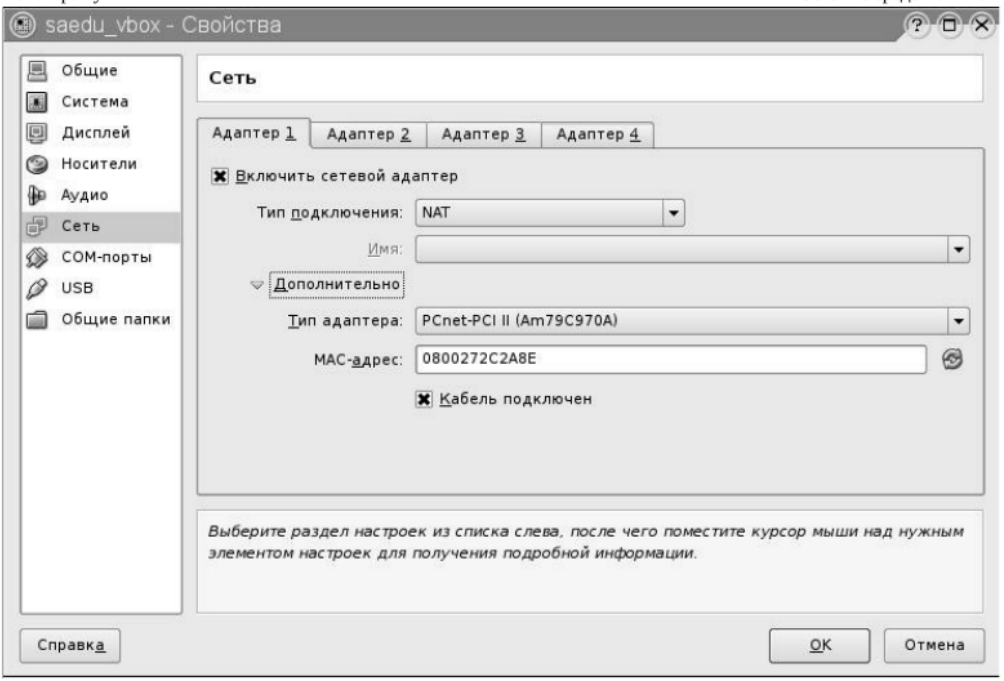

Рис. 1.5. Настройка сети ВМ

Каждый сетевой адаптер может быть настроен на один из следующих режимов работы:

- Режим "Не подключен" (Not attached). В данном режиме адаптер присутствует в гостевой системе, но ведет себя так, как будто сетевой кабель в него не включен.
- Режим "NAT". В данном режиме адаптер использует сетевые настройки основной системы при взаимодействии с сетью физического узла и прочими внешними сетями. Сетевая подсистема VirtualBox транслирует IP-трафик с исходным IP-адресом виртуальной машины в трафик с исходным адресом сетевого адаптера хостовой системы (трансляция сетевых адресов, Network Address Translation). Реализация NAT в VirtualBox имеет определенные ограничения, связанные с поддержкой протокола ICMP, широковещательного UDP -трафика и технологий виртуальных частных сетей.
- Режим "Сетевой мост" (Bridged networking). В данном режиме сетевой адаптер ВМ подключается к сетевому адаптеру хостовой системы и обрабатывает сетевые пакеты непосредственно в обход

сетевого стека хостовой системы (адаптер хостовой системы работает с адаптером ВМ в режиме моста).

- Режим "Внутренняя сеть" (Internal Networking). Данная внутренняя сеть представляет собой физический сегмент, объединяющий включенные в него виртуальные машины, причем внутренняя сеть не имеет доступа к основной системе и внешним сетям.
- Режим "Виртуальный адаптер хоста" (Host-only networking). Сеть объединяющая в данный сегмент хостовую систему и виртуальные машины, включенные в данный сегмент. Для этого режима VirtualBox создает в хостовой системе программный сетевой интерфейс и устанавливает на нем IP-адрес.

В режимах NAT и виртуального адаптера хоста VirtualBox предоставляет виртуальным машинам сервис динамической настройки протокола IP (DHCP — Dynamic Host Configuration Protocol). Соответствующий сетевой адаптер, настроенный в ВМ на динамическую конфигурацию в данных режимах будет получать IP-параметры автоматически от DHCP -сервера VirtualBox.

Упражнения

Используя образ установочного диска Debian (предпочтительно DVD, так как содержит весь необходимый софт и не потребуется использование Интернет-репозиториев) создать виртуальный компьютер на основе Linux:

1. Используя мастер создания новой виртуальной машины, создать виртуальную машину в VirtualBox: указать имя ВМ, операционная система: Linux, версия: Debian, размер памяти: 256 Мбайт; создать новый загрузочный жесткий диск размером 800 Мбайт.
2. В свойствах созданной виртуальной машины подключить образ загрузочного установочного DVD-диска (или CD-диска) Debian.
3. Запустить виртуальную машину. С диска запустится программа установки Debian.
4. В процессе установки Debian:
    - выбрать русский язык и регион (меню "Choose language");
    - выбрать раскладку клавиатуры (меню "Выбор раскладки клавиатуры");

- позволить инсталлятору настроить сеть автоматически (IP-адрес будет выдан DHCP-сервером NAT-интерфейса VirtualBox);
- указать имя компьютера;
- указать домен;
- выбрать ручной метод разметки дисков (меню "Разметка дисков");
- выбрать диск SCSI1 (0,0,0) (sda) и создать пустую таблицу разделов на нем;
- создать два первичных раздела: раздел подкачки (128 Мбайт) и корневой раздел (точка монтирования – "/");
- закончить разметку и записать изменения на диск;
- установить пароль суперпользователя;
- создать непривилегированного пользователя testuser и установить пароль для него;
- Затем программа установки инсталлирует базовую систему.
- в меню "Выбор программного обеспечения" оставить отмеченным пункт "Стандартная система";
- выбрать способ переключения раскладки клавиатуры и временный переключатель (меню "Cyrillic on Console");
- установить системный загрузчик GRUB в главную загрузочную запись;
- перезагрузиться в установленную систему.

Примечание: В дальнейшем для установки через apt пакетов с DVD-диска, его необходимо предварительно примонтировать:

```
# mount -t iso9660 /dev/cdrom /media/cdrom
```

при этом, в /etc/apt/sources.list должно быть указано что-то вроде:

```
deb cdrom:[Debian GNU/Linux 6.0.0 _Squeeze_ - Official i386 DVD Binary-1
  20110205-17:27]/ squeeze contrib main
```

Сетевая подсистема Linux

Кратко описан многоуровневый иерархический подход к обеспечению сетевых взаимодействий. Перечислены базовые средства настройки и диагностики сетевой подсистемы Linux.

Содержание:

Многоуровневый подход к организации сетевых взаимодействий

Процесс взаимодействия между узлами компьютерной сети является сложным "Транспортный и прикладной уровни модели сетевого взаимодействия.". Сложности связаны как с преобразованием "данные-сигналы-данные", так и с передачей сигнала по гетерогенным сетям. Для наглядной иллюстрации процесса передачи данных в сети Интернет можно, например, с помощью команды `traceroute` (tracert в Windows), изучить список промежуточных узлов, через которые проходят сетевые пакеты на пути от компьютера пользователя до какого-нибудь Интернет-ресурса (пример 2.1)

```
$ traceroute www.kernel.org
 traceroute to www.kernel.org (130.239.17.4), 30 hops max, 60 byte packets
  1  192.168.240.1 (192.168.240.1)  78.016 ms  77.871 ms  77.795 ms
  2  * * *
  3  * * *
  4  core.10g.net.belpak.by (93.84.122.105)  109.257 ms  109.196 ms  109.13
  5  core.10g.net.belpak.by (93.84.122.18)  117.289 ms  117.227 ms  117.164
  6  193.232.249.102 (193.232.249.102)  108.876 ms  12.377 ms  12.257 ms
  7  po31.l3-gw-1.mck.net.belpak.by (193.232.249.78)  22.113 ms  16.070 ms
  8  * * *
  9  83.229.226.101 (83.229.226.101)  55.643 ms msk-b4-hq-ae0.main.syntern
       (83.229.225.5)  55.571 ms 83.229.226.101 (83.229.226.101)  55.512 ms
 10  ae7.RT.MR.MSK.RU.retn.net (87.245.253.169)  55.450 ms  55.388 ms  5
```

```
11  ae2-6.RT.TC1.STO.SE.retn.net (87.245.233.134)  70.087 ms  70.025 ms
12  * * *
13  t1fre-ae0-v1.sunet.se (130.242.83.37)  98.493 ms umu-g.sunet.se (193.11.
     99.859 ms t1fre-ae0-v1.sunet.se (130.242.83.37)  90.245 ms
14  umu-g.sunet.se (193.11.0.226)  112.213 ms u2k-te.gw.umu.se (130.239.0.
     112.181 ms umu-g.sunet.se (193.11.0.226)  112.137 ms
15  pub4.kernel.org (130.239.17.4)  112.108 ms u2k-te.gw.umu.se (130.239.0
     117.516 ms  117.392 ms
```

Листинг 2.1. Трассировка маршрута (отображение списка промежуточных узлов) до узла www.kernel.org с помощью команды traceroute:

Задачу обеспечения сетевых коммуникаций решают с использованием многоуровневого иерархического подхода. На узле программные компоненты данного уровня "общаются" только с соседними уровнями, при этом интерфейсы (предоставляемые функции) между соседними уровнями четко определены. Узлы взаимодействуют друг с другом посредством протоколов (правил, по которым обмениваются сообщениями программные компоненты одного уровня на разных узлах) "Использование виртуализации для изучения Linux". Для передачи по сети поток информации на узле-отправителе разбивается на набор фрагментов (единиц передачи данных), по мере продвижения по иерархии протоколов, каждый протокол добавляет (инкапсулирует) свой заголовок к фрагменту передаваемой информации. На узле-получателе происходит обратный процесс.

Существует несколько альтернативных моделей сетевых взаимодействий "Использование виртуализации для изучения Linux". В вопросах настройки сетевой подсистемы (в отличие от вопросов разработки протоколов и интерфейсов) значение данных моделей в большей степени методическое, направленное на унификацию терминологии и лучшее понимание процессов функционирования сети.

В данном пособии используется терминология (нестрогая) модели стека протоколов TCP/IP, которая включает четыре уровня:

- Уровень сетевых интерфейсов (доступа к сети). В рамках модели TCP/IP на данном уровне решаются две задачи: упаковки единиц передачи вышележащего уровня во фреймы (единицы передачи

данного уровня) и получения физических адресов (MAC-адресов).

- Сетевой уровень (интернет-уровень, межсетевой уровень). Обеспечивает передачу пакетов (единиц передачи сетевого уровня) между гетерогенными сетями на основе IP-адресации (сетевой адресации) и группировки узлов в подсети. Основным протоколом данного уровня является IP (Internet Protocol).
- Транспортный уровень. Обеспечивает для фрагментов данных как гарантированную доставку (протокол TCP, единицы передачи - сегменты), так и доставку по возможности (протокол UDP, единицы передачи - датаграммы). На данном уровне решается задача использования сетевого подключения несколькими приложениями (порты).
- Прикладной уровень. Включает средства преобразования и протоколы представления информационных потоков, непосредственно взаимодействующие с приложениями пользователя.

Пример 2.2. Типичный процесс взаимодействия между сетевыми узлами по прикладному протоколу, использующему в качестве транспорта протокол TCP, можно описать следующим образом на узле-отправителе:

- На прикладном уровне осуществляются необходимые преобразования информационного потока (сжатие, шифрование и т. д.).
- Информационный поток прикладного протокола на транспортном уровне разбивается на последовательность сегментов; в заголовке сегмента, в частности, указаны номера портов отправителя и получателя.
- На сетевом уровне сегмент вкладывается в пакет, заголовок, которого содержит IP-адреса отправителя и получателя.
- На уровне сетевых интерфейсов на основе сетевого пакета формируется фрейм, заголовок которого содержит MAC-адреса отправителя и получателя (в случае "нелокальной" доставки - маршрутизатора удаленных сетей).

На узле-получателе происходит обратный процесс.

Примечание: В некоторых случаях на сетевом уровне и уровне сетевых интерфейсов используется дополнительная инкапсуляция (технологии глобальных сетей, аутентификации, виртуальные частные сети и т.п.)

Средства настройки сетевой подсистемы Linux

Сетевая поддержка в Linux обеспечивается стандартными и дополнительными модулями ядра, программным обеспечением, входящим в состав конкретного дистрибутива, а также может быть расширена за счет использования прочих программных средств "Доступ к локальной сети средствами Linux" "Команды настройки протокола IP".

Для управления сетевой подсистемой Linux существует множество средств: команды, скрипты, графические программы и т. д. Здесь приведены лишь наиболее известные из них.

Для базовой настройки сетевой подсистемы можно использовать следующие группы команд:

"Традиционные" команды настройки сети (пакет net-tools) "Постоянные сетевые конфигурации (на примере Debian/GNU Linux)". В данном пакете присутствуют команды: управления сетевыми интерфейсами (`ifconfig`), таблицей маршрутизации (`route`), таблицей разрешения имен (`arp`), сетевой статистики (`netstat`), статусом сетевых устройств (`mii-tool`) и т.д. Пакет net-tools использовался для управления сетью до ядра 2.4.; включается в современные дистрибутивы Linux для совместимости.

Команды пакета iproute2 "Базовая диагностика сетевых подключений". Пакет iproute2 разработан для управления параметрами TCP/IP и контроля трафика в Linux (в частности маршрутизация и качество обслуживания). Для управления сетью в современных ядрах Linux (ядра версии 2.4-2.6) рекомендуется использовать средства данного пакета вместо команд `net-tools`.

Пакет iproute2 включает 3 основные утилиты:

`-ip` - команда для просмотра параметров и настройки сетевых интерфейсов, IP-адресов, таблиц маршрутизации, правил

маршрутизации, таблиц преобразования, IP-туннелей и т.д.

`-tc` (traffic control) - команда для просмотра и настройки параметров управления трафиком (классификация трафика, дисциплины управления очереди для различных классов трафика)

`-ss` - команда для просмотра текущих соединений и открытых портов (аналог `netstat`).

Для настройки сети при загрузке системы (а также как средство настройки работающей системы) в Linux обычно используются наборы конфигурационных файлов и обрабатывающих их скриптов и программ. В дистрибутивах применяются различные способы инициализации сетевой подсистемы, например Debian использует пакет ifupdown (см. "Постоянные сетевые конфигурации (на примере Debian/GNU Linux)"), а большинство RPM -основанных дистрибутивов — набор файлов вида ifup-*, ifdown-*, ifcfg-*, расположенных в каталоге /etc/sysconfig/network-scripts/ и т.п.

В системах с установленным оконным менеджером для настройки сетевых параметров можно использовать средства графического интерфейса пользователя. В качестве примера можно назвать пакет NetworkManager "Транспортный и прикладной уровни модели сетевого взаимодействия.", совместимый с таким графическими окружениями, как GNOME, KDE и XFCE ; интерфейс NetworkManager позволяет настраивать проводные и беспроводные (Wi-Fi, 3G, Bluetooth) соединения, подключения к виртуальным частным сетям (VPN) и ADSL -провайдерам. Средства графического интерфейса пользователя есть также в пакете Wicd [14].

Для управления межсетевым экраном и средством манипулирования сетевыми пакетами (в том числе трансляция сетевых адресов - NAT) Netfilter стандартно используется утилита командной строки iptables [10] (см. "Межсетевое экранирование в Linux" и "Обеспечение доступа в сеть Интернет"). Над Netfilter/iptables существуют различные надстройки (в том числе и GUI -средства), например, UFW/Gufw — простые утилиты настройки межсетевого экрана уровня узла (host-based), используемые в дистрибутиве Ubuntu [9]. Примерами программ для настройки межсетевого экрана также могут служить такие проекты как: Shorewall

[9], Firewall Builder [11], Firestarter [12] и т. д.

В состав дистрибутивов Linux обычно также включаются разнообразные средства диагностики и мониторинга сети. Простейшими и общеупотребительными из них являются:

- Команда `ping` [13]. Простейшее средство проверки работоспособности удаленного узла.
- Команда *`traceroute`*. Позволяет отобразить все промежуточные узлы, которые проходят пакеты на пути к целевому узлу.
- Утилита `tcpdump`. Используется для перехвата и анализа сетевого трафика.

Более подробно эти и некоторые другие средства диагностики сети описаны в "Базовая диагностика сетевых подключений" и "Анализ сетевого трафика как метод диагностики сети".

Ключевые термины

Гетерогенная сеть - информационная сеть, в которой работают различные протоколы, используются технологии и оборудование различных фирм-производителей.

Интерфейс - формализованные правила, в соответствии с которыми взаимодействуют модули, реализующие протоколы соседних уровней модели сетевого взаимодействия (набор сервисов, предоставляемых данным уровнем соседнему).

Протокол - формализованные правила, определяющие последовательность и формат сообщений, которыми обмениваются сетевые компоненты, лежащие на одном уровне модели сетевого взаимодействия в разных узлах.

IP (Internet Protocol) — маршрутизируемый протокол сетевого уровня без установления соединения.

Сокет — пара IP-адрес: порт, однозначно определяющая сетевой процесс; точка доступа прикладного процесса к сети.

Краткие итоги

1. К решению задачи обеспечения сетевых взаимодействий применяют многоуровневый иерархический подход, заключающийся в разбиении процесса коммуникации на набор уровней с четко определенными способами взаимодействия уровней на одном узле и на соседних узлах.
2. В данном пособии используется терминология модели стека протоколов TCP/IP, включающая следующие уровни: уровень сетевых интерфейсов, сетевой уровень, транспортный уровень и прикладной уровень.
3. Для управления сетевой подсистемой Linux существует множество средств, наиболее известными из которых являются средства пакетов net-tools, iproute2, Netfilter/iptables, утилиты диагностики сетевых подключений (`ping`, *`traceroute`*, `tcpdump`), а также графические средства настройки сетевой подсистемы (например, NetworkManager).

Доступ к локальной сети средствами Linux

Описаны базовые аспекты взаимодействия компьютеров в сети Ethernet. Приведены примеры команд настройки параметров сетевого адаптера в Linux на уровне сетевых интерфейсов.

Содержание:

Взаимодействие компьютерных систем в сети Ethernet

Группа технологий Ethernet [20], [7] предназначена для организации взаимодействия компьютеров на уровне сетевых интерфейсов (определяет характеристики среды передачи, типы соединений, способы коммуникации устройств и адресации, используемое оборудование и т. д.); чаще всего используется для построения локальных сетей.

В качестве среды передачи в настоящее время в Ethernet используется витая пара (несколько пар изолированных проводников, скрученных между собой) или оптоволоконный кабель. Полоса пропускания различных технологий Ethernet варьирует от 10 Мбит/с до 10 Гбит/с. Все новые стандарты Ethernet обратно совместимы с базовым стандартом (условно говоря, единица передачи данных - Ethernet-фрейм передается в любой среде без изменений). При этом, оборудование различных производителей, использующее различные технологии Ethernet, обычно, без проблем работает в одной сети.

Для подключения компьютера к среде передачи (витая пара) используется сетевой адаптер с разъемом RJ-45. В данном сегменте сети каждый сетевой адаптер имеет уникальный физический адрес (MAC-адрес), размером 48 бит, который принято записывать в виде последовательности шестнадцатеричных цифр (например, `00:1d:d9:43:0f:41`). Принцип действия сетевого адаптера — в том, что обрабатываются те Ethernet-фреймы, у которых указан его

MAC-адрес в адресе получателя (а также специальные широковещательные фреймы, у которых MAC-адрес равен `FF:FF:FF:FF:FF:FF`).

Сетевой адаптер компьютера подключаются к коммутатору — многопортовому устройству, предназначенному для пересылки Ethernet-фреймов между портами. Коммутатор ведет таблицу коммутации (MAC-адресов), описывающую соответствие между MAC-адресами узлов и портами, к которым они подключены. Записи в таблице MAC-адресов имеют определенное время жизни, по истечению которого они удаляются из таблицы.

Для передаваемых фреймов коммутатор анализирует MAC-адреса

1. Для передаваемых фреймов коммутатор анализирует MAC-адрес отправителя и получателя и содержимое таблицы коммутации (рис. 3.1.):
2. Если MAC-адрес отправителя отсутствует в таблице, то он вносится туда с привязкой к порту.
3. Если MAC-адреса отправителя и получателя привязаны к одному и тому же порту, то фрейм не пересылается (фильтрация).
4. Если MAC-адреса отправителя и получателя привязаны к различным портам, то фрейм отправляется в порт получателя.
5. Если MAC-адрес получателя неизвестен коммутатору, производится широковещательная рассылка фрейма по всем портам, за исключением порта источника.
6. Фреймы, у которых в адресе получателя указан широковещательный адрес (`FF:FF:FF:FF:FF:FF`) также рассылаются по всем портам, кроме исходного.

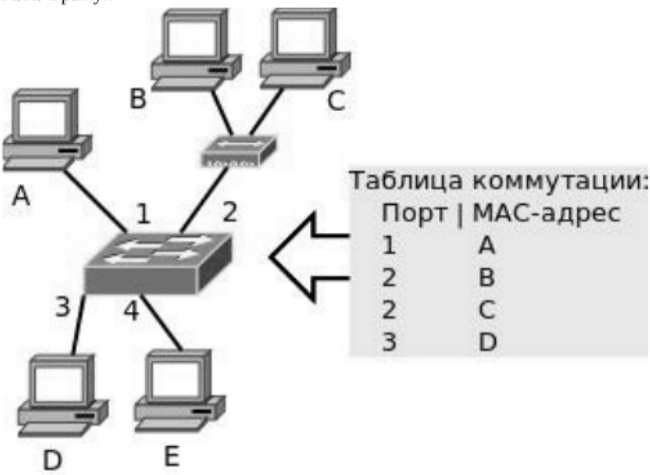

Рис. 3.1. Коммутатор и таблица коммутации

Управление сетевыми интерфейсами в Linux.

Физический сетевой интерфейс — устройство, способное отправлять в физическую сеть и получать из сети атомарные единицы передачи — фреймы; предоставляется драйвером устройства и имеет определенное имя, которое используется для доступа к нему (например, драйвера Ethernet ядро именует как eth0, eth1 и т.д.). Фактически, чтобы использовать сетевой интерфейс, нужно настроить на нем какой-либо протокол сетевого уровня (например, протокол IP).

Для управления сетевыми интерфейсами на уровне доступа к сети можно использовать команду `ip` из пакета iproute2 с параметром `link` [22].

```
# ip link show
 1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKI
     link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq
     qlen 1000 link/ether 00:1d:72:01:ab:93 brd ff:ff:ff:ff:ff:ff
 3: irda0: <NOARP> mtu 2048 qdisc noop state DOWN qlen 8
     link/irda 00:00:00:00 brd ff:ff:ff:ff
 4: vboxnet0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOW
     qlen 1000 link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
 5: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 14(
```

```
    pfifo_fast state UNKNOWN qlen 3 link/ppp
```

Листинг 3.1. Вывод информации о сетевых интерфейсах (ip link show):

Вывод данной команды содержит пронумерованный список интерфейсов, присутствующих в системе. Информация об интерфейсе содержит две строки: в первой указывается имя интерфейса, установленные флаги состояния, MTU (Maximum Transmission Unit, максимально допустимый размер фрейма в байтах), тип и размер очереди фреймов; во второй строке — тип соединения, MAC-адрес, широковещательный адрес и т. п.

Некоторые флаги состояния:

- `UP` — устройство подключено и готово принимать и отправлять фреймы;
- *`LOOPBACK`* — интерфейс является локальным и не может взаимодействовать с другими узлами в сети;
- *`BROADCAST`* - устройство способно отправлять широковещательные фреймы;
- `POINTTOPOINT` — соединение типа "точка-точка"
- `PROMISC` — устройство находится в режиме "прослушивания" и принимает все фреймы.
- `NOARP` — отключена поддержка разрешения имен сетевого уровня.
- `ALLMULTI` — устройство принимает все групповые пакеты.
- `NO-`*`CARRIER`* — нет связи (не подключен кабель).
- `DOWN` — устройство отключено.

Можно также вывести информацию о данном интерфейсе, задав его имя в параметре `dev`:

```
# ip link show dev eth0
```

или просто:

```
# ip link show eth0
```

Примечание:У параметров команд пакета iproute2 есть сокращенный

синтаксис, например, `ip link show eth0` можно записать как `ip l sh eth0`.

Команда `ip link` также позволяет менять некоторые свойства сетевого интерфейса. Для этого используется параметр `set`.

```
# ip link set mtu 1400 eth1
 # ip link show eth1
 3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1400 qdisc pfif
    UP qlen 1000 link/ether 08:00:27:66:6a:4f brd ff:ff:ff:ff:ff:ff
```

Листинг 3.2. Уменьшение MTU сетевого интерфейса с помощью ip link set mtu:

Примечание: Все произведенные изменения командами, описываемые в данном разделе, будут действительны до перезагрузки компьютера или перезапуска сетевой подсистемы скриптами инициализации. Как редактировать постоянные сетевые конфигурации, описано в "Постоянные сетевые конфигурации (на примере Debian/GNU Linux)". "Постоянные сетевые конфигурации".

```
# ip link set address 00:11:11:12:FE:09 eth1
 # ip link show eth1
 3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1400 qdisc pfif
    UP qlen 1000 link/ether 00:11:11:12:fe:09 brd ff:ff:ff:ff:ff:ff
```

Листинг 3.3. Смена MAC-адреса сетевого интерфейса с помощью ip link set address

Используя `ip link`, можно включить или отключить сетевой интерфейс пример 3.4

```
# ip link set down eth1
 # ip link set up eth1
```

Листинг 3.4. Выключение (down) и включение (up) сетевого интерфейса с помощью ip link

Ниже приводится пример использования команды `ifconfig` из пакета net-tools для управления сетевым интерфейсом пример 3.5.

```
# ifconfig eth1 down
 # ifconfig eth1 up
 # ifconfig eth1 mtu 1400
 # ifconfig eth1 hw ether 00:11:12:13:14:15
 # ifconfig eth1
 eth1      Link encap:Ethernet  HWaddr 00:11:12:13:14:15
           inet6 addr: fe80::a00:27ff:fe66:6a4f/64 Scope:Link
           UP BROADCAST RUNNING MULTICAST  MTU:1400  Metric:1
           RX packets:0 errors:0 dropped:0 overruns:0 frame:0
           TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:0 (0.0 B)  TX bytes:706 (706.0 B)
```

Листинг 3.5. Использование ifconfig для управления сетевым интерфейсом

Настройка физических параметров сетевого подключения

Физические параметры (скорость, технология Ethernet, тип дуплекса) сетевого подключения зависят от используемого оборудования и обычно настраиваются автоматически при подключении компьютера к сети.

В отдельных случаях несогласованной работы оборудования и драйверов адаптера параметры можно выставить вручную, используя утилиту `ethtool`. Данная утилита включена в стандартные репозитории Debian; для ее установки достаточно выполнить команду:

```
#apt-get install ethtool

# ethtool eth1
 Settings for eth1:
   Supported ports: [ TP ]
   Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full
   Supports auto-negotiation: Yes
   Advertised link modes:  10baseT/Half 10baseT/Full
```

```
                    100baseT/Half 100baseT/Full
                    1000baseT/Full
    Advertised auto-negotiation: Yes
    Speed: 1000Mb/s
    Duplex: Full
    Port: Twisted Pair
    PHYAD: 0
    Transceiver: internal
    Auto-negotiation: on
    Supports Wake-on: umbg
    Wake-on: d
    Current message level: 0x00000007 (7)
    Link detected: yes
```

Листинг 3.6. Просмотр параметров сетевого интерфейса eth1 с помощью ethtool

Вывод содержит информацию о поддерживаемых технологиях Ethernet, автоматическом определении сети, текущей скорости передачи, дуплексе и т. д.

```
# ethtool -i eth1
 driver: e1000
version: 7.3.20-k2-NAPI
 firmware-version: N/A
 bus-info: 0000:00:08.0
```

Листинг 3.7. Вывод сведений об используемых драйверах с помощью ethtool (параметр -i)

```
# ethtool -s eth1 speed 10 duplex half
```

Листинг 3.8. Установка (параметр -s) скорости подключения 10 Мбит/с (параметр speed) и передачи в режиме полудуплекса (параметр duplex)

Ключевые термины

Локальная сеть (LAN - Local Area Network) — высокоскоростная сеть передачи данных, занимающая небольшую территорию.

Ethernet — группа технологий уровня доступа к сети, использующих в качестве среды передачи витую пару и оптоволоконный кабель.

Физический адрес (MAC-адрес, Media Access Control) — 48-битный номер, присваиваемый сетевому адаптеру и используемый для адресации на уровне доступа к сети.

Коммутатор — многопортовое устройство уровня доступа к сети, осуществляющее пересылку Ethernet - фреймов между портами на основе информации, извлекаемой из заголовков фреймов, а также поддерживаемых таблиц, определяющих направление пересылки (таблиц коммутации).

Дуплекс — режим работы приемопередающего устройства, при котором устройство может передавать и получать информацию одновременно.

Краткие итоги

1. Группа технологий Ethernet на основе витой пары и оптоволоконного кабеля широко используется для построения локальных (и не только) сетей.
2. Сетевой адаптер обычно обрабатывает те фреймы, в адресе получателя которых указан его адрес или широковещательный адрес, а все остальные игнорирует.
3. Коммутатор Ethernet в рамках широковещательного домена согласно собственной таблице коммутации пересылает фреймы между узлами, подключенными к его портам.
4. Для управления физическими интерфейсами применятся команда `ip link` из пакета iproute2 (справку по синтаксису данной команды, а также других можно получить с помощью подсистемы помощи, например man ip). Также сетевые параметры можно изменять с помощью команды `ifconfig` пакета net-tools.
5. Для настройки характеристик физического подключения (скорость, технология, тип дуплекса используется команда `ethtool`.

Упражнения

В среде VirtualBox запустите виртуальную машину с установленным

Debian GNU/Linux с одной сетевой картой, предварительно установив тип сетевого подключения "Виртуальный адаптер хоста" (см. "Использование виртуализации для изучения Linux") и выполните следующие задания:

1. Определить MAC-адрес сетевого адаптера виртуальной машины.
2. Отключите сетевой интерфейс.
3. Включите сетевой интерфейс.
4. Установите другой MAC-адрес для сетевого адаптера.

Команды настройки протокола IP

Приведены базовые сведения об IP-адресации, механизме сетевых масок, протоколе ARP и маршрутизации. Рассмотрены примеры использования команд ip, ifconfig, arp, route для управления IP-адресами и ARP -таблицей, а также добавления шлюза по умолчанию и альтернативных шлюзов.

Содержание:

Протокол IP как средство организации межсетевого взаимодействия

Сетевой уровень модели сетевого взаимодействия обеспечивает доставку пакетов для узлов составной сети [1] (совокупности подсетей, построенных на основе различных технологий уровня интерфейсов и использующих различные типы магистральных каналов для взаимодействия).

Основой сетевого уровня является интернет-протокол IP (Internet Protocol), имеющий следующие ключевые особенности:

1. IP внедряет данные вышележащих уровней в соответствующие единицы передачи информации - сетевые пакеты;
2. IP является протоколом без установления соединения, т. е. не требует физического или логического канала и не гарантирует доставку пакетов;
3. IP представляет собой маршрутизируемый протокол. Организация протокола IP обеспечивает схему доставки пакетов на основе IP-адресации и группировки сетевых узлов в подсети.

Каждый IP-пакет помимо данных вышележащего уровня содержит заголовок, в котором указаны IP-адреса отправителя и получателя, а также описан ряд параметров: тип службы, идентификационную информацию, номер протокола вышележащего уровня, параметры, используемые при фрагментации, контрольную сумму и т.д. (более подробные сведения приведены в [7] [23]). Длина IP-заголовка составляет 20 байт, максимальная длина пакета (MTU) — 64Кбайт (реальная длина пакета ограничена возможностями передающей среды).

IP-адрес (сетевой адрес) протокола IP версии 4 (IPv4) представляет собой 4-х байтовый номер, который принято записывать в десятичной форме, разделяя байты точкой (например: 192.168.4.3). Схема адресации составной сети (в т.ч. сети Интернет) является независимой от способов адресации узлов в отдельных сетях [7]. Сетевой адрес объединяет в себе информацию об адресе сети и адресе узла в данной сети.

Для решения задачи определения сетевой части IP-адреса (количество бит в IP-адресе, отведенное для записи адреса сети, обычно располагаемое в "начале") наиболее распространен механизм сетевых масок (двоичная запись маски содержит единицы в тех разрядах, которые должны в IP-адресе интерпретироваться как номер сети).Маска может записываться в формате, подобном IP-адресу, а также путем указания количества бит, отведенных для адреса сети (например, `255.255.255.0` то же самое, что и `/24`). Результатом операции логического умножения IP-адреса и маски будет адрес сети, которой принадлежит данный узел.

Примеры интерпретации IP-адресов в зависимости от маски сети:

`192.168.92.0/24` — сеть, включающая узлы от `1` до `254` (`0` — адрес сети; `255` — зарезервирован для широковещательных рассылок, каждый последний узел в сети).

`192.168.92.5 255.255.255.0` — узел в сети `192.168.92.0`.

`192.168.92.130/25` — узел в сети `192.168.92.128`, которая включает узлы от `129` до `254`.

Манипулируя масками, можно объединять и разделять диапазон IP-

адресов. Например, сеть `192.168.9.0/24` может быть разбита на две подсети путем увеличения сетевой части адреса на 1 бит: `192.168.9.0/25` (узлы `1-126`) и `192.168.9.128/25` (узлы `129-254`).

Примечание: Существуют программные средства (в т.ч. online) для автоматического расчета подсетей на основе масок, например, сайт [24] и соответствующая программа ipcalc, которая присутствует в репозиториях Debian.

Управление адресацией сети Интернет осуществляется по иерархической схеме (региональные центры выдачи IP-адресов - национальные центры — Интернет-провайдеры — клиенты - субклиенты и т. д.). В качестве координатора выступает некоммерческая организация ICANN (Internet Corporation for Assigned Names and Numbers). Для организации локальных (автономных) сетей (не имеющих непосредственного доступа к сети Интернет) используются зарезервированные диапазоны IP-адресов: `10.0.0.0/8`, `172.16.0.0-172.31.0.0/12`, `192.168.0.0/16`.

Управление IP-адресами в Linux

Для того, чтобы сетевой протокол мог использовать физический интерфейс, необходимо добавить соответствующий адрес (настроить логический интерфейс). В случае стека TCP/IP минимальными настройками протокола IP, позволяющими узлу взаимодействовать с другими узлами подсети в рамках локального сегмента (широковещательного домена) являются IP-адрес и маска подсети.

Для конфигурирования протокола IP на сетевом интерфейсе с помощью командной строки можно использовать команду `address (addr)` утилиты `ip` пакета `iproute2` или традиционную команду `ifconfig`.

Примечание: Все произведенные изменения командами, описываемые в данном разделе, будут действительны до перезагрузки компьютера или перезапуска сетевой подсистемы скриптами инициализации. Как редактировать постоянные сетевые конфигурации, описано в "Постоянные сетевые конфигурации"

Для просмотра списка IP-адресов используется команда `ip addr show` [24] [25] [26]. Вывод команды содержит список сетевых интерфейсов, их параметров, а также список IP-адресов (пример 4.1.) на каждом интерфейсе. Можно вывести информацию только по данному интерфейсу, указав его имя (например, `ip addr show eth0`).

```
$ ip addr show
 1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKI
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq
    link/ether 00:1d:72:01:ab:93 brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.11/24 brd 192.168.11.255 scope global eth0
    inet6 fe80::21d:72ff:fe01:ab93/64 scope link
       valid_lft forever preferred_lft forever
 3: irda0: <NOARP> mtu 2048 qdisc noop state DOWN qlen 8
    link/irda 00:00:00:00 brd ff:ff:ff:ff
 4: vboxnet0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOW
    link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
 5: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 14(
    pfifo_fast state UNKNOWN qlen 3 link/ppp inet 10.31.46.4 peer 172.20.2(
    scope global ppp0
```

Листинг 4.1. Отображение списка IP-адресов с использованием команды ip addr show

Каждая запись о сетевом адресе может содержать информацию о типе адреса (`inet` — адрес *IPv4*, `inet6` — *IPv6* и т. д.), непосредственно адрес, маску подсети (количество сетевых бит), широковещательный адрес данной сети, область видимости (`scope global` — действителен везде, `scope link` — только для данного устройства, `scope host` — для данного узла) и имя логического интерфейса.

Сведения подобного рода также предоставляет команда `ifconfig`, дополнительно отображая статистику работы сетевого интерфейса

(пример 4.2).

```
$ ifconfig eth0
 eth0   Link encap:Ethernet  HWaddr 00:1d:72:01:ab:93
        inet addr:192.168.11.11  Bcast:192.168.11.255  Mask:255.255.255.0
        inet6 addr: fe80::21d:72ff:fe01:ab93/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:69333 errors:0 dropped:0 overruns:0 frame:0
        TX packets:69736 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:37148192 (37.1 MB)  TX bytes:5619320 (5.6 MB)
        Interrupt:16
```

Листинг 4.2. Вывод информации о сетевом интерфейсе eth0 с помощью ifconfig

Для удаления IP-адреса используется команда `ip addr del` пример 4.3. При этом необходимо указать удаляемый IP-адрес (`192.168.11.11`), маску подсети `(/24)` и имя интерфейса (`dev eth0`). Изменение параметров сети требует полномочий суперпользователя.

```
# ip address del 192.168.11.11/24 dev eth0
 # ip addr show eth0
 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq
    link/ether 00:1d:72:01:ab:93 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::21d:72ff:fe01:ab93/64 scope link
      valid_lft forever preferred_lft forever
```

Листинг 4.3. Удаление IP-адреса с интерфейса eth0 с помощью ip addr del

Для добавления IP-адреса используется команда `ip addr add` (пример 4.4.). При этом необходимо указать добавляемый IP-адрес, маску подсети, имя интерфейса, а также широковещательный адрес (в данном случае автоматически рассчитать широковещательный адрес на основе маски: *`brd +`*, можно также указать адрес вручную: *`brd`* `192.168.11.255`).

```
# ip address add 192.168.11.11/24 brd + dev eth0
```

Листинг 4.4. Добавление IP-адреса на интерфейс eth0 с использованием ip addr add

Добавить IP-адрес можно также с помощью команды `ifconfig` (пример 4.5).

```
# ifconfig eth0 inet up 192.168.11.11 netmask 255.255.255.0
```

Листинг 4.5. Добавление IP-адреса на интерфейс eth0 с помощью ifconfig

Несколько IP-адресов на одном сетевом интерфейсе

В некоторых случаях необходимо использовать несколько IP-адресов на одном сетевом интерфейсе [25].

Пусть на сетевом интерфейсе eth1 установлен IP-адрес 192.168.11.10/24, необходимо, чтобы система принимала на этом интерфейсе входящие соединения на адрес 192.168.11.11. Добавим на интерфейс еще один адрес:

```
# ip addr add 192.168.11.11/24 brd + dev eth1
# ip addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo
    link/ether 08:00:27:66:6a:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.10/24 brd 192.168.11.255 scope global eth1
    inet 192.168.11.11/24 brd 192.168.11.255 scope global secondary eth1
    inet6 fe80::a00:27ff:fe66:6a4f/64 scope link
       valid_lft forever preferred_lft forever
```

Листинг 4.6. Добавление дополнительного IP-адреса

При добавлении адреса из той же сети, что и существующий, новый адрес становится дополнительным к существующему (secondary), и если удалить основной адрес, то будет удален и дополнительный.

Примечание: Команда `ifconfig` непосредственно не поддерживает несколько IP-адресов на одном сетевом интерфейсе (только в виде псевдонимов). Поэтому secondary - адрес в выводе `ifconfig` отображаться не будет:

```
# ifconfig eth1
eth1   Link encap:Ethernet  HWaddr 08:00:27:66:6a:4f
       inet addr:192.168.11.10  Bcast:192.168.11.255  Mask:255.255.255.0
       inet6 addr: fe80::a00:27ff:fe66:6a4f/64 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:74 errors:0 dropped:0 overruns:0 frame:0
       TX packets:74 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:8640 (8.4 KiB)  TX bytes:7020 (6.8 KiB)
```

Примечание: Для удаления всех сетевых адресов на интерфейсе можно использовать командуip addr *flush*:

```
# ip addr flush dev eth1
```

```
# ip addr add 192.168.11.11/24 brd + dev eth1 label eth1:add
# ip addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo
    link/ether 08:00:27:66:6a:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.10/24 brd 192.168.11.255 scope global eth1
    inet 192.168.11.11/24 brd 192.168.11.255 scope global secondary eth1:add
# ifconfig
eth1   Link encap:Ethernet  HWaddr 08:00:27:66:6a:4f
       inet addr:192.168.11.10  Bcast:192.168.11.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:756 errors:0 dropped:0 overruns:0 frame:0
       TX packets:711 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:75800 (74.0 KiB)  TX bytes:68550 (66.9 KiB)
eth1:add Link encap:Ethernet  HWaddr 08:00:27:66:6a:4f
       inet addr:192.168.11.11  Bcast:192.168.11.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Листинг 4.7. Для реализации ifconfig - совместимого способа использования нескольких IP-адресов на одном сетевом интерфейсе следует в команде ip addr add использовать параметр label

Управление ARP-таблицей

В сегменте сети Ethernet для взаимодействия устройств друг с другом по протоколу IP необходимо, чтобы у передающего устройства были логический (IP-) и физический (MAC-) адреса получателя. Для решения задачи определения MAC-адреса узла назначения по его IP-адресу в локальном сегменте используется протокол ARP (Address Resolution Protocol) [7] .Принцип действия протокола ARP заключается в том, что отправитель сетевого пакета посылает широковещательный ARP-запрос (на широковещательный MAC-адрес FF:FF:FF:FF:FF:FF) с указанием искомого IP-адреса. ARP-запрос получают все устройства в сегменте, но только тот, чей IP-адрес является искомым, отправляет ARP-ответ (на MAC-адрес отправителя) с указанием своего MAC-адреса.

Каждый узел локальной сети поддерживает ARP-таблицу (кэш), которая устанавливает соответствие между IP и MAC-адресами узлов подсетей, в которые входит данный узел. В ARP-таблице имеются два вида записей:

- Динамические записи, которые периодически обновляются с использованием протокола ARP (если запись "устаревает", то она удаляется).
- Статические записи, которые создаются пользователем с помощью соответствующих команд и существуют до тех пор, пока узел не будет выключен.

Используя команду `ip neigh` или традиционную команду `arp` можно манипулировать содержимым ARP-таблицы.

Пример 4.8. Просмотр ARP-таблицы.

Если узел 192.168.11.10 через интерфейс eth1 осуществит сетевое взаимодействие с узлом 192.168.11.12 (например, с помощью утилиты ping), то в его ARP-таблице появится новая запись, которую можно просмотреть командой `ip neigh show` (параметр `dev` указывает фильтровать записи относящиеся к интерфейсу eth1):

```
# ping -c 1 192.168.11.12
PING 192.168.11.12 (192.168.11.12) 56(84) bytes of data.
64 bytes from 192.168.11.12: icmp_seq=1 ttl=64 time=1.58 ms
--- 192.168.11.12 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.588/1.588/1.588/0.000 ms
 #  ip neigh show dev eth1
192.168.11.12 lladdr 08:00:27:23:22:97 REACHABLE
```

Пример 4.9. Добавление статической записи в ARP-таблицу.

Для добавления статической записи в ARP-таблицу можно использовать команду `ip neigh add` с указанием IP-адреса, MAC-адреса (параметр `lladdr`) и сетевого интерфейса (параметр `dev`):

```
# ip neigh add 192.168.11.100 lladdr 00:00:00:00:00:AA dev eth1
# ip neigh show dev eth1
192.168.11.100 lladdr 00:00:00:00:00:aa PERMANENT
```

Для удаления записи из ARP-таблицы можно использовать команду `ip neigh del`:

```
# ip neigh del 192.168.11.100 dev eth1
```

```
# arp -s 192.168.11.100 00:00:00:00:00:AA
# arp -i eth1
Address  HWtype  HWaddress  Flags Mask Iface
192.168.11.100     ether   00:00:00:00:00:aa   CM  eth1
# arp -d 192.168.11.100
```

Листинг 4.10. Использование команды arp для манипулирования ARP-таблицей (ключ -s — добавить статическую запись, -i — фильтр интерфейса, -d— удалить запись)

Управление маршрутами

Доставку сетевых пакетов в рамках локального сегмента сети называют локальной.

Пример 4.11. Локальная доставка пакетов

IP-пакет с заголовком 192.168.9.5 > 192.168.9.10 (рис. 4.1) будет доставлен локально (фрейм отправляется непосредственно на MAC-адрес узла 10, т. к. согласно маске /24 узел 10 находится в одной подсети

с узлом-отправителем).

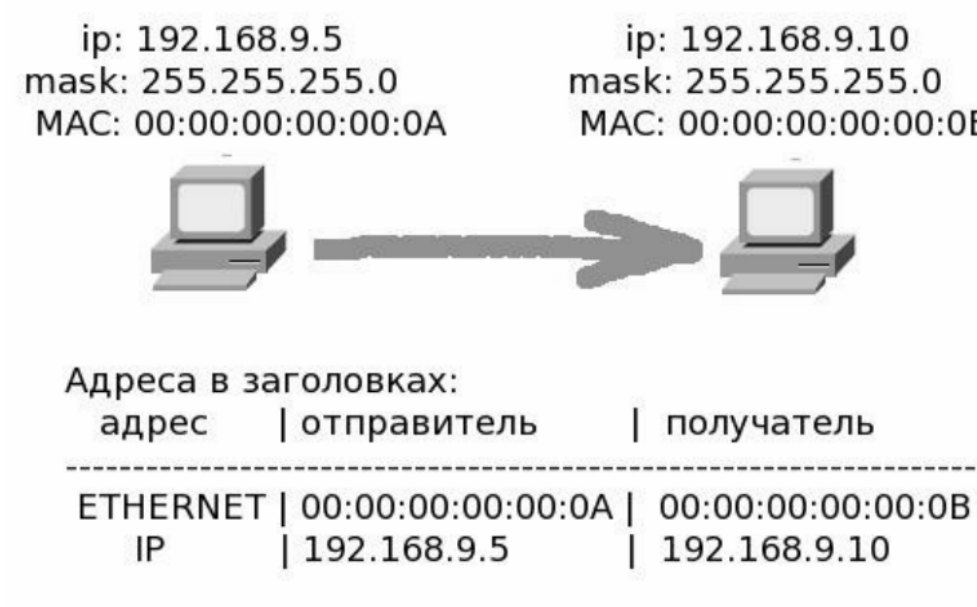

Рис. 4.1. Локальная доставка сетевого пакета

Стандартный шлюз (шлюз по умолчанию) - IP-адрес интерфейса узла данной подсети, способный отправлять и принимать пакеты удаленных подсетей для данной подсети. Все IP-пакеты, предназначенные для узлов из других подсетей, данный узел отправляет на MAC-адрес стандартного шлюза ("нелокальная" доставка).

Пример 4.12. "Нелокальная" доставка пакетов

Сетевой пакет 192.168.9.5 /24 > 192.168.10.100 (рис. 4.2) будет отправлен на MAC-адрес шлюза данной подсети (шлюз 192.168.9.1 — должен быть указан в настройках данного узла как шлюз по умолчанию).

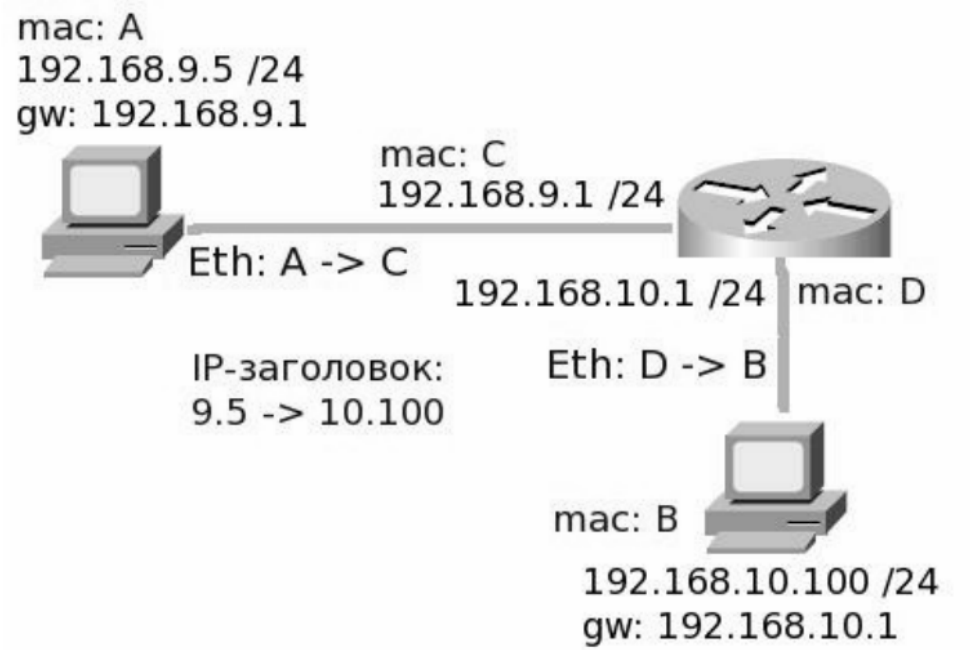

Рис. 4.2. Использование шлюза по умолчанию

Для отдельных сетей и узлов можно указывать маршруты через альтернативные шлюзы. При этом, все шлюзы должны иметь IP-адреса из той же подсети, что и передающий узел, а также находится в одном широковещательном домене с передающим узлом (сеть Ethernet).

Маршрутизация на узле (решение на какой шлюз отправлять пакет, предназначенный узлу "нелокальной" сети) осуществляется на основе таблицы маршрутизации.

Пример 4.13. Добавление альтернативного шлюза.

Пусть в локальной сети 192.168.1.0/24 есть шлюз сети Интернет 192.168.1.1 (использующий, например трансляцию сетевых адресов - NAT) и шлюз удаленной сети 192.168.24.0/24, имеющий адрес 192.168.1.2 (рис.4.3).

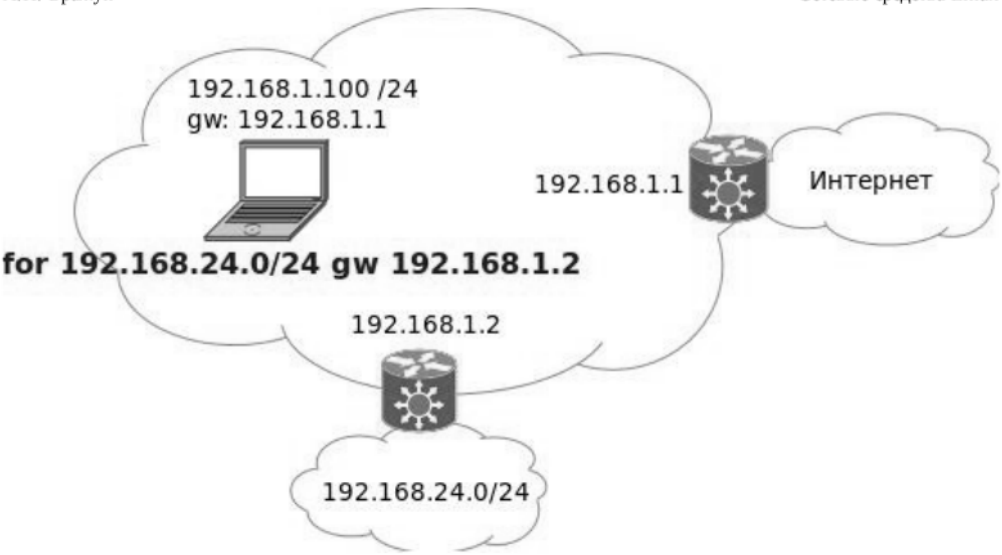

Рис. 4.3. Использование маршрута по умолчанию и альтернативного маршрута

Для того, чтобы компьютер с адресом 192.168.1.100/24 (рис. 4.3) имел доступ в сеть Интернет и к локальной сети 192.168.24.0/24, необходимо внести в таблицу маршрутизации соответствующие записи, например с помощью команды `ip route add`:

```
# ip route add default via 192.168.1.1
# ip route add 192.168.24.0/24 via 192.168.1.2
# ip route show
192.168.1.0/24 dev eth1  proto kernel  scope link  src 192.168.1.100
192.168.24.0/24 via 192.168.1.2 dev eth1
default via 192.168.1.1 dev eth
```

Первая команда добавляет маршрут по умолчанию (`default`) через узел 192.168.1.1 (параметр `via`). Вторая команда устанавливает маршрут на сеть 192.168.24.0/24 через узел 192.168.1.2.

Для вывода на экран содержимого таблицы маршрутизации используется команда `ip route show`. В данном случае в таблице маршрутизации три записи: первая о том, что сеть 192.168.1.0/24 доступна непосредственно на интерфейсе eth1 (запись добавляется автоматически), и две записи, добавленные пользователем:

альтернативный маршрут и маршрут по умолчанию.

Те же действия (см. выше) можно выполнить, используя команду `route`:

```
# route add default gw 192.168.1.1
# route add -net 192.168.24.0 netmask 255.255.255.0 gw 192.168.1.2
# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.24.0    192.168.1.2     255.255.255.0   UG    0      0        0 eth1
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth
```

Примечание: При нескольких альтернативных маршрутах, возможных для данного узла выбирается тот, у которого маска подсети "более заполнена" пример 4.14.

```
# ip route show

192.168.10.3 via 192.168.56.2 dev eth0

192.168.10.0/25 via 192.168.56.3 dev eth0

192.168.10.0/24 via 192.168.56.2 dev eth0

192.168.56.0/24 dev eth0  proto kernel  scope link  src 192.168.56.101

default via 192.168.56.1 dev eth0
```

Листинг 4.14. Выбор альтернативного маршрута

В данном примере есть несколько перекрывающихся маршрутов (1-3 записи). В соответствии с "заполненностью" маски, пакеты для узла 192.168.10.3 будут отправляться шлюзу 192.168.56.2, т. к. у этой записи самая "большая" маска (/32). Пакеты остальных узлов сети 192.168.10.0/25 будут отправляться шлюзу 192.168.56.3, так как у второй записи в таблице маршрутизации маска (/25) больше, чем у третьей (/24). Пакеты, предназначенные узлам 128-255 сети 192.168.10.0/24 будут отправляться шлюзу 192.168.56.2.

Ключевые термины

IP (Internet Protocol) — маршрутизируемый протокол сетевого уровня без установления соединения.

IP-адрес (сетевой адрес) — адрес, используемый протоколом IP для реализации схемы адресации в рамках составной сети.

Маска подсети — способ указания сетевой части IP-адреса, основанный на записи единиц в тех битах IP-адреса, которые указывают на адрес сети.

ARP (Address Resolution Protocol) — протокол, решающий задачу определения MAC-адреса по его IP-адресу в рамках сегмента сети (широковещательного домена).

ARP-таблица (кэш) — таблица соответствия IP-адресов узлов данной подсети их MAC-адресам, поддерживаемая узлом.

Маршрутизация — доставка сетевого пакета получателю по оптимальному маршруту.

Стандартный шлюз (шлюз по умолчанию) - IP-адрес интерфейса узла данной подсети, способный отправлять и принимать пакеты удаленных подсетей для данной подсети.

Краткие итоги

1. Протокол IP является маршрутизируемым протоколом без установления соединения. IP адрес представляет собой комбинацию номера сети и номера узла в данной сети. Количество бит, используемых для номера сети определяется по классу адреса или задается битовой маской.
2. Для того, чтобы узел мог взаимодействовать с другими узлами подсети в рамках локального сегмента (широковещательного домена) на нем должен быть установлен IP-адрес и определена маска подсети. Для управления IP-адресами в Linux можно использовать команду `ip addr` или традиционную команду `ifconfig` (подробный синтаксис и параметры большинства

команд Linux можно посмотреть с помощью справочной подсистемы man, например: man ifconfig).

3. В локальной сети данный узел для определения физических адресов по IP - адресам других узлов использует протокол ARP и локальный ARP -кэш (таблица). Управление ARP -таблицей осуществляется командой `ip neigh` или `arp`.
4. Для взаимодействия с удаленными подсетями на данном узле необходимо определить шлюз по умолчанию или/и альтернативные шлюзы. Все шлюзы должны находиться в той же подсети, что и исходный узел. Управление таблицей маршрутизации на узле может осуществляться командой `ip route` или `route`.

Упражнения

1. Не используя дополнительных средств [24], разделите сеть 192.168.9.0/24 на 4 подсети. Указать адреса подсетей, сетевую маску и диапазоны номеров узлов, входящих в каждую подсеть.
2. В среде VirtualBox запустите виртуальную машину с установленным Debian GNU/Linux с одной сетевой картой, предварительно установив тип сетевого подключения "Виртуальная сеть узла" (см. "Использование виртуализации для изучения Linux") и выполните следующие задания:
   - Определите IP-адрес сетевого адаптера виртуальной машины (выдается виртуальной машине средой VirtualBox по протоколу динамической конфигурации узла - DHCP). Определите, какой шлюз по умолчанию настроен в виртуальной машине.
   - Определите MAC-адрес шлюза по умолчанию.
   - .Установите дополнительный IP-адрес с номером на 5 больше основного.
   - .Удалите дополнительный IP-адрес.
3. В среде VirtualBox запустите две виртуальные машины с установленным Debian GNU/Linux с одной сетевой картой у каждой, настроенной на тип сетевого подключения "Внутренняя сеть". Установите на первой виртуальной машине IP-адрес 192.168.9.5/24, а на второй - IP-адрес 192.168.10.158/27. Добавьте необходимые дополнительные адреса на виртуальные машины,

таким образом, чтобы они могли взаимодействовать друг с другом, используя основные IP-адреса (проверку доступности узла осуществлять командой: `ping IP_адрес_удаленного_компьютера` или `ping -I локальный_IP_адрес IP_адрес_удаленного_компьютера`

4. Если узел 1 находится в одной IP-сети с узлом 2 и не отвечает на ARP-запросы, то взаимодействие узла 2 с узлом 1 возможно, если:
    - Вручную внести в таблицу преобразования адресов узла 2 IP- и MAC-адрес узла 1.
    - Вручную внести в таблицу преобразования адресов узла 1 IP- и MAC-адрес узла 2.
    - В таблице маршрутизации узла 1 присутствует маршрут по умолчанию.
    - Взаимодействие возможно без дополнительных условий.
5. Узлы 1 и 2 находятся в одной IP-сети, но не могут взаимодействовать. Укажите наиболее вероятную причину:
    - Не настроен маршрут по умолчанию на одном из узлов.
    - На одном из узлов правила брандмауэра запрещают сетевые взаимодействия.
    - IP-сеть, в которую входят узлы, является автономной.
6. Узлы 1 и 2 находятся в одной IP-сети (узел 1 имеет IP-адрес 192.168.8.5/24, узел 2 — 192.168.8.6/24). На узле 2 дополнительно установлен IP-адрес 192.168.10.100. Что необходимо сделать на узле 1, чтобы пакеты с адреса 192.168.8.5 доходили до адреса 192.168.10.100?
    - Добавить альтернативный маршрут к узлу 192.168.10.100 через узел 192.168.8.6.
    - Внести в таблицу преобразования адресов MAC-адрес, соответствующий узлу 192.168.8.6.
    - Внести в таблицу преобразования адресов MAC-адрес, соответствующий узлу 192.168.10.100.
7. На компьютере A установлены следующие настройки протокола IP: IP-адрес 192.168.9.12 /24, шлюз по умолчанию 192.168.12.1. Компьютер успешно взаимодействует с компьютером B, имеющим IP-адрес 192.168.10.5. Почему MAC-адрес компьютера B не отображается в таблице преобразования адресов компьютера A?

Постоянные сетевые конфигурации (на примере Debian/GNU Linux)

Даны общие сведения о пакете ifupdown, используемом в Debian/GNU Linux для автоматизации настройки сетевых параметров. Приведены примеры настройки интерфейсов, дополнительного маршрута, нескольких IP-адресов и виртуальных локальных сетей (VLAN) с помощью конфигурационного файла /etc/network/interfaces.

Содержание:

Автоматическая инициализация сетевой подсистемы

Пакет ifupdown является комплексным средством настройки параметров сети, в частности используется для инициализации сети при загрузке операционной системы в Debian GNU\Linux (совместно с соответствующими скриптами `ifupdown-clean`, `ifupdown`, `networking`, расположенными в каталоге /etc/init.d) [28].

Примечание: Начиная с версии 6.0 (squeeze), разработчики Debian в документации классифицируют пакет ifupdown как устаревший и рекомендуют пользоваться такими средствами как NetworkManager или Wicd. Такой подход себя оправдывает для рабочих станций с установленными средствами графического интерфейса пользователя. Для серверов продолжается использование ifupdown - стабильного, многофункционального, хорошо документированного средства управления сетевой подсистемой.

Пакет ifupdown содержит две команды `ifup` и `ifdown` для включения и отключения сетевого подключения (пример 5.1). Данные команды по

умолчанию используют настройки, записанные в файле /etc/network/interfaces.

```
# ifdown eth1
# ifup eth1
```

Листинг 5.1. Использование ifdown и ifup для отключения и включения сетевого интерфейса eth1

Для запуска, перезапуска и останова сетевой подсистемы следует использовать скрипт `/etc/init.d/networking` с параметрами `start`, *`restart`* *и* `stop` соответственно (пример 5.2).

```
# /etc/init.d/networking restart
```

Листинг 5.2. Перезапуск сети с помощью /etc/init.d/networking

Файл настроек сетевых интерфейсов /etc/network/interfaces

Файл /etc/network/interfaces имеет текстовый формат, пригодный для редактирования администратором системы с помощью текстового редактора, в тоже время команды `ifup` и `ifdown` также способны его читать и распознавать указанные в нем настройки.

```
auto lo eth1 eth0
iface lo inet loopback

iface eth1 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    gateway 192.168.1.1
    dns-nameservers 192.168.1.1

iface eth0 inet dhcp
```

Листинг 5.3. Файл /etc/network/interfaces

Ключевое слово `auto` (пример 5.3) с перечислением через пробел имен

интерфейсов указывает включать данные интерфейсы при старте системы. Ключевое слово `iface` является описанием интерфейса (общий формат: `iface <имя_интерфейса> <тип адреса> <метод_настройки>`). Так:

1. в строке `iface lo inet` *`loopback`* настраивается локальный интерфейс lo для взаимодействия приложений в рамках данного компьютера (loopback).
2. в строке `iface eth1 inet static` настраивается интерфейс eth1 на статический метод конфигурации (сетевые параметры указываются вручную, static). Далее параметры статической настройки интерфейса и указаны (формат в общем виде: `<опция> <значение>`): IP-адрес (`address`), маска подсети (*`netmask`*), шлюз по умолчанию (`gateway`), адреса DNS-серверов (`dns-`*`nameservers`*) и т.д.
3. в строке `iface eth0 dhcp` указывается настройка интерфейса eth0 по протоколу динамической конфигурации узла (dhcp).

Протокол DHCP (Dynamic Host Configuration Protocol) позволяет компьютеру автоматически получать по сети IP-адрес и другие параметры, необходимые для работы сетевого интерфейса. Для использования протокола DHCP необходимо чтобы в данном широковещательном домене был настроен DHCP-сервер. При настройке сетевого устройства компьютер обращается к DHCP-серверу и получает требуемые сетевые параметры.

Примечание. Кроме статической и динамической настройки интерфейса, существует ручной способ конфигурирования (manual), который предполагает то, что интерфейс будет настроен с помощью средств, сторонних относительно `ifupdown`.
Примечание. Рекомендуется отключать сетевой интерфейс (`ifdown`) перед тем как вносить изменения в его конфигурацию, а затем включать (`ifup`).
Примечание. Синтаксис файла `interfaces` подробно описан на соответствующей странице помощи (`man interfaces`).

Добавление постоянного статического маршрута

Опции интерфейса в файле interfaces позволяют указать команды, которые необходимо выполнить при включении интерфейса (опция `up`) и выключении (опция `down`). В частности данные опции можно использовать для настройки дополнительных маршрутов (пример 5.4.).

Настроим постоянную сетевую конфигурацию для решения задачи, описанной в примере 4.13 (предыдущая лекция), с добавлением статического маршрута через альтернативный шлюз (приведен фрагмент файла /etc/network/interfaces):

```
iface eth1 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    up ip route add 192.168.24.0/24 via 192.168.1.2
    gateway 192.168.1.1
```

Листинг 5.4. Добавление постоянного статического маршрута

Применим данную конфигурацию с помощью команд `ifdown/ifup` и просмотрим таблицу маршрутизации:

```
# ifdown eth1
# ifup eth1
# ip route show dev eth1
192.168.1.0/24  proto kernel  scope link  src 192.168.1.100
192.168.24.0/24 via 192.168.1.2
default via 192.168.1.1
```

Несколько IP-адресов на одном сетевом интерфейсе

Для решения задачи, описанной в примерах 4.6. и 4.7. (предыдущая лекция) отредактируем файл /etc/network/interfaces следующим образом (приведен отрывок файла):

```
auto eth1 eth1:add
iface eth1 inet static
    address 192.168.11.10
    netmask 255.255.255.0
    gateway 192.168.11.1
```

```
iface eth1:add inet static
    address 192.168.11.11
    netmask 255.255.255.0
```

Листинг 5.5. Добавление дополнительного IP-адреса

В данном случае псевдоним задается через двоеточие после имени интерфейса. Также необходимо указать автоматический запуск нового интерфейса в параметре `auto`. Для применения данной конфигурации необходимо выполнить последовательность команд:

```
# ifdown eth1; ifup eth1; ifup eth1:add
```

или

```
# /etc/init.d/networking restart
```

Несколько виртуальных локальных сетей (VLAN) на одном интерфейсе

Технология виртуальных локальных сетей (Virtual Local Area Network, VLAN) позволяет в рамках одной физической коммутируемой сети выделить несколько изолированных на канальном уровне виртуальных сетей [7] . Для организации VLAN необходимо, чтобы телекоммуникационное оборудование (коммутаторы) поддерживало эту функцию; для взаимодействия виртуальных локальных сетей друг с другом используют средства сетевого уровня (маршрутизация).

Существуют два способа группировки компьютеров сети в виртуальные сети [7]:

- Группировка портов коммутатора. Каждый порт коммутатора сопоставляют соответствующей виртуальной сети.
- Группировка MAC-адресов. При данном подходе каждый MAC-адрес в сети должен быть прописан в той или иной виртуальной сети.

Независимо от способа группировки коммутатор коммутирует фреймы

(в том числе и широковещательные) в рамках VLAN и не коммутирует их между VLAN.

Для передачи данных между коммутаторами фреймы соответствующих виртуальных сетей маркируются (тегируются), т. е. в заголовке Ethernet-фрейма указывается номер виртуальной сети. Соответствующие порты, соединяющие коммутаторы друг с другом называются транковыми. В современных сетях для пересылки VLAN-трафика между коммутаторами используется протокол IEEE 802.1Q (могут встречаться также фирменные протоколы, совместимые с оборудованием данного производителя).

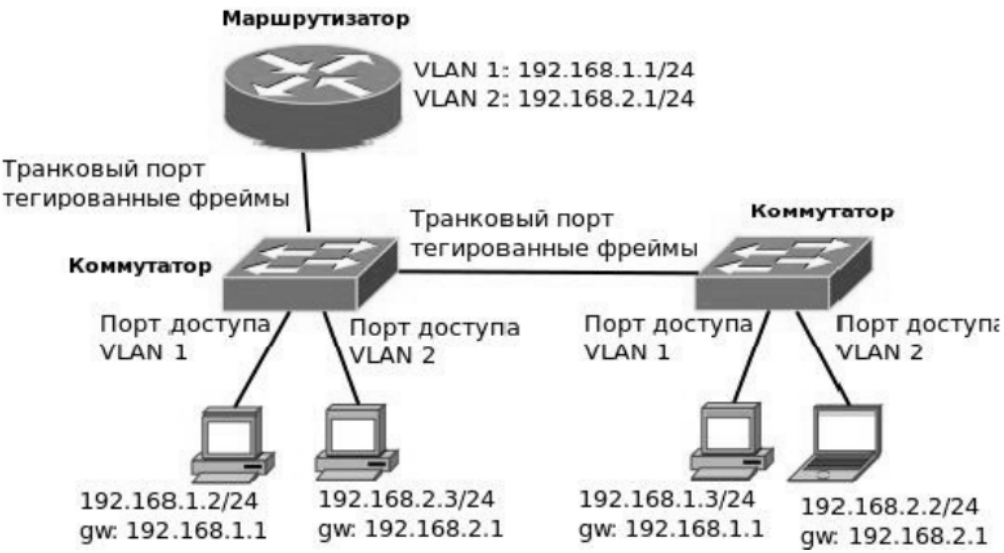

Рис. 5.1. Пример использования группировки портов

На рис. 5.1. изображен пример построения виртуальной локальной сети на основе группировки по портам. Порты, к которым подключены рабочие станции, настроены на коммутаторах как порты доступа, т. е. каждому порту доступа сопоставлен номер VLAN (1 или 2); при этом фреймы на этих портах "нетегированы". Коммутаторы и маршрутизатор соединены транковыми портами, на которых фреймы каждого VLAN нумеруются соответственно. На коммутаторах сети VLAN 1 и VLAN 2 изолированы друг от друга. Их взаимодействие происходит на маршрутизаторе на сетевом уровне (если фильтры маршрутизатора позволяют такое взаимодействие); для этого первой виртуальной сети

сопоставлена IP-подсеть 192.168.1.0/24 и в ней маршрутизатор имеет адрес 192.168.1.1, вторая сеть — 192.168.2.0/24 и адрес интерфейса маршрутизатора - 192.168.2.1.

Ядро Linux поддерживает тегированные фреймы стандарта 802.1Q (пример 5.6). Для настройки VLAN используется утилита vconfig из пакета vlan. Ее необходимо установить из стандартных репозиториев Debian:

```
# apt-get install vlan
```

Пусть требуется подключить компьютер с Linux в рамках схемы, изображенной на рис. 5.1, таким образом чтобы он имел доступ как к первой виртуальной сети, так и второй. Для этого, предварительно необходимо настроить порт коммутатора, к которому подключается данный компьютер, как транковый. Затем в файле /etc/network/interfaces указать (приведен фрагмент файла):

```
auto vlan1
iface vlan1 inet static
  address 192.168.1.4
  netmask 255.255.255.0
  vlan_raw_device eth1

auto vlan2
iface vlan2 inet static
  address 192.168.2.4
  netmask 255.255.255.0
  vlan_raw_device eth1
```

Листинг 5.6. Настройка виртуальных частных сетей на сетевом интерфейсе

где, номер виртуальной сети определяется в строке описания интерфейса (`vlan1` — первый, `vlan2` — второй), а физический интерфейс определяется опцией `vlan_raw_device` (в данном случае `eth1`).

Для применения изменений перезапустим сетевую подсистему:

```
# /etc/init.d/networking restart
```

Отрывок вывода команды `ip addr show` для данной конфигурации:

```
4: vlan1@eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qd
   link/ether 08:00:27:66:6a:4f brd ff:ff:ff:ff:ff:ff
   inet 192.168.1.4/24 brd 192.168.1.255 scope global vlan1
   inet6 fe80::a00:27ff:fe66:6a4f/64 scope link
     valid_lft forever preferred_lft forever
5: vlan2@eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qd
   link/ether 08:00:27:66:6a:4f brd ff:ff:ff:ff:ff:ff
   inet 192.168.2.4/24 brd 192.168.2.255 scope global vlan2
   inet6 fe80::a00:27ff:fe66:6a4f/64 scope link
     valid_lft forever preferred_lft foreve
```

Ключевые термины

DHCP (Dynamic Host Configuration Protocol) - протокол для автоматической настройки сетевых параметров с использованием специального сервера DHCP.

DNS (Domain Name System) — иерархическая система, позволяющая преобразовать символьные имена узлов и ресурсов Интернет в IP-адреса и обратно.

Виртуальная локальная сеть (VLAN, Virtual LAN) — изолированный сегмент сети уровня сетевых интерфейсов, не привязанный к физическому расположению узлов.

Краткие итоги

1. Пакет ifupdown представляет собой комплексное средство настройки сетевых подключений в Linux (в частности Debian GNU\Linux). Основными компонентами пакета являются команды `ifup` и `ifdown`, а также конфигурационный файл /etc/network/interfaces.
2. Файл /etc/network/interfaces можно редактировать с помощью любого текстового редактора (имея права суперпользователя).

Секция описания интерфейса iface позволяет задать статический (`static`), динамический (`dhcp`) или ручной (с помощью команд, *`manual`*) способ настройки интерфейса. Опции настройки интерфейсов дают возможность указать основные характеристики интерфейса, установить дополнительные IP-адреса, выполнить заданные действия при включении и выключении интерфейса и т. д.

3. Linux поддерживает работу с тегированными фреймами Ethernet стандарта 802.1Q на уровне ядра. Для управления параметрами виртуальных локальных сетей используется утилита vconfig из пакета vlan. Соответствующую конфигурацию сетевого интерфейса можно описать в файле /etc/network/interfaces.

Упражнения

1. Изучите справочное руководство по файлу interfaces (man interfaces) и укажите какие опции используются для установки MAC-адреса и MTU при статической конфигурации интерфейса.
2. В среде VirtualBox запустите виртуальную машину с установленным Debian GNU/Linux с одной сетевой картой, предварительно установив тип сетевого подключения "Виртуальная сеть узла" (см. "Использование виртуализации для изучения Linux") и настройте сетевой интерфейс локальной сети с помощью файла /etc/network/interfaces на статическую конфигурацию с IP-адресом 192.168.10.12/16, шлюзом по умолчанию и DNS-сервером 192.168.1.1, MTU для интерфейса установите 1400. Опишите дополнительный IP-адрес 192.168.10.100 из той же подсети, что и основной. Также опишите альтернативный маршрут до сети 10.30.0.0/16 через узел 192.168.254.1. Активируйте указанную конфигурацию и проверьте ее состояние с помощью команды ip.
3. Для упражнения №3 предыдущей лекции (Команды настройки протокола IP) опишите файлы /etc/network/interfaces для первой и второй виртуальной машины.
4. Настройте на сетевом интерфейсе виртуальной машины виртуальные интерфейсы для работы с тегированным трафиком VLAN с номерами 4, 5 и 6.

Базовая диагностика сетевых подключений

Даны общие сведения о протоколе ICMP. Приведены примеры использования приложений ICMP — утилиты ping и traceroute и пример использования утилиты netstat для получения статистики сетевых подключений.

Содержание:

Основы протокола ICMP

Протокол ICMP (Internet Control Message Protocol) используется для диагностики сетевых подключений, а также для передачи удаленным системам сообщений об исключительных ситуациях, возникающих в процессе работы стека TCP/IP. ICMP работает на сетевом уровне модели сетевого взаимодействия; при этом ICMP инкапсулирует свои сообщения в IP-пакеты; реализация ICMP является обязательной для стека TCP/IP. ICMP-сообщение состоит из заголовка (содержит идентификаторы типа и кода сообщения, а также контрольную сумму) и поля данных, содержимое которого зависит от назначения сообщения [7].

Условно ICMP -сообщения можно разделить на две группы [30]: сообщения об ошибках и информационные сообщения. Первая группа включает такие типы сообщений как: направление недоступно (Destination Unreachable); подавление источника (Source Quench); истечение времени (Time Exceeded); проблема параметров (Parameter Problem) и т.д. Примером сообщений второй группы являются сообщения перенаправление (Redirect); эхо-запроса (Echo) и эхо-ответа (Echo Reply). Всего известно около 40 типов ICMP -сообщений.

Большинство типов сообщений содержат наборы уточняющих кодов, конкретизирующих проблему. Так сообщение "Направление недоступно" информирует о недоступности удаленного узла по той или иной причине: недоступность узла, протокола или порта; большой размер пакета по сравнению с MTU передающей среды при невозможности фрагментации пакетов; отсутствие у промежуточного шлюза маршрутов к удаленному узлу или сети; активные фильтры пакетов на шлюзе и т.д.

Пусть на узле A установлен IP-адрес из сети 192.168.1.0/24 и шлюз по умолчанию 192.168.1.1. Если, например шлюз "не знает" маршрута до сети 192.168.2.0/24, то при попытке взаимодействия узла A с узлом 192.168.2.10 (например, с помощью утилиты `ping`), узел A будет получать от шлюза ICMP -сообщения о недоступности сети назначения (Destination Net Unreachable).

```
# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
From 192.168.1.1 icmp_seq=1 Destination Net Unreachable
From 192.168.1.1 icmp_seq=2 Destination Net Unreachable
^C
--- 192.168.2.10 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 1023ms
```

Листинг 6.1. ICMP-сообщение о недоступности сети

В данном примере один из шлюзов фильтрует доступ к сети Интернет и отправляет соответствующие ICMP-сообщения узлу локальной сети.

```
$ ping www.yandex.ru
PING www.yandex.ru (213.180.204.3) 56(84) bytes of data.
From 10.31.46.1 icmp_seq=1 Packet filtered
From 10.31.46.1 icmp_seq=2 Packet filtered
^C
--- www.yandex.ru ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 10074ms
```

Листинг 6.2. ICMP-сообщение о фильтрации пакетов

Примечание. Функциональность ICMP на данном узле является настраиваемой. Например, настройки маршрутизатора позволяют указать какое ICMP -сообщение отправлять на данную ошибку

протокола IP, и отправлять ли вообще какие-либо сообщения.

Проверка доступности удаленной системы с помощью команды ping

Самый простой способ тестирования связи между двумя системами заключается в проверке возможности отправки системами сообщений друг другу. Для этого первый узел отправляет второму узлу тестовое сообщение, получив которое второй узел должен отправить ответное сообщение. В протоколе ICMP для отправки тестового сообщения используется сообщение типа эхо-запрос (`Echo`), для отправки ответа на тестовое сообщение — эхо-ответ (`Echo Reply`). Функция ответа на эхо-запросы интегрирована в сетевой стек операционной системы и в большинстве случаев включена по умолчанию. Для реализации данного типа проверки используется утилита `ping`, которая в Debian устанавливается по умолчанию (пример 6.3.). Основным параметром `ping` является доменное имя удаленной системы или ее IP-адрес.

```
$ ping www.yandex.ru
PING www.yandex.ru (87.250.250.3) 56(84) bytes of data.
64 bytes from www.yandex.ru (87.250.250.3): icmp_req=1 ttl=52 time=42.8 ms
64 bytes from www.yandex.ru (87.250.250.3): icmp_req=2 ttl=52 time=42.5 ms
64 bytes from www.yandex.ru (87.250.250.3): icmp_req=3 ttl=52 time=50.0 ms
^C
--- www.yandex.ru ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 42.514/45.124/50.044/3.481 ms
```

Листинг 6.3. Проверка доступности удаленной системы

В Linux команда `ping` по умолчанию отправляет эхо-запросы до тех пор, пока пользователь не прервет ее работу, например комбинацией клавиш Ctrl+C (можно использовать параметр `-c` с указанием количества отправляемых запросов, например: `ping -c 3 www.yandex.ru` - будут отправлены 3 запроса). Если ответ на запрос получен, то на экран выводится информация о размере эхо-ответа (в данном случает 64 байт), доменное имя узла и(или) IP-адрес, порядковый номер запроса, параметры TTL и RTT (см. ниже). При

завершении своей работы команда `ping` выводит статистику: количество отправленных (*transmitted*) и полученных (*received*) пакетов, процент потерь (*packet loss*), а также статистику *RTT*.

Параметр TTL (Time-To-Live) определяет время "жизни" пакета и на узле-источнике устанавливается в целое число (максимальное количество маршрутизаторов, которое может быть пройдено пакетом - скачков). При каждом скачке значение TTL уменьшается на единицу. Когда значение становится равным нулю, пакет уничтожается. Данный подход позволяет избежать петель маршрутизации (бесконечного перемещения пакета между двумя узлами маршрутизации), возникающих, например, при ошибках конфигурации.

Примечание. Различные системы первоначальное значение TTL устанавливают в различные значения (255 – сетевое оборудование Cisco, 128 – Windows, Unix-системы — 64, некоторое сетевое оборудование — 30), благодаря чему в некоторых случаях в первом приближении можно определить тип удаленной системы.

Параметр RTT (Round Trip Time) — время между отправкой запроса и получением ответа на данный запрос.

Информация, предоставляемая командой ping, позволяет сделать определенные выводы о качестве канала связи между двумя узлами. Например, высокий процент потерь пакетов или меняющееся в больших пределах значение *RTT* являются указанием на наличи проблем со связью. В тоже время на основе хороших показателей потерь пакетов и *RTT* не всегда можно сделать заключение о пригодности канала связи для использования данным транспортным или прикладным протоколом, т. к. они могут предъявлять более высокие требования к качеству канала связи. Сравните, например, значение параметр *RTT* для коротких пакетов и пакетов, размер которых близок к максимальному возможному (пример 6.4.) в сегменте сети Ethernet.

```
# ping -s 1400 -c 3 10.31.16.103
PING 10.31.16.103 (10.31.16.103) 1400(1428) bytes of data.
1408 bytes from 10.31.16.103: icmp_seq=1 ttl=64 time=2.17 ms
1408 bytes from 10.31.16.103: icmp_seq=2 ttl=64 time=1.90 ms
```

```
1408 bytes from 10.31.16.103: icmp_seq=3 ttl=64 time=1.26 ms
--- 10.31.16.103 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 1.260/1.779/2.171/0.384 ms
```

Листинг 6.4. Использование ключа -s для указания размера тестовых сообщений в команде ping

Для сравнения ping в тех же условиях с малым размером сообщения:

```
# ping -c 3 10.31.16.103
PING 10.31.16.103 (10.31.16.103) 56(84) bytes of data.
64 bytes from 10.31.16.103: icmp_seq=1 ttl=64 time=0.328 ms
64 bytes from 10.31.16.103: icmp_seq=2 ttl=64 time=0.231 ms
64 bytes from 10.31.16.103: icmp_seq=3 ttl=64 time=0.583 ms
--- 10.31.16.103 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.231/0.380/0.583/0.150 ms
```

Выбор исходного IP-адреса для отправки пакетов в системах с несколькими IP-адресами зависит от настроек таблиц маршрутизации и особенностей операционной системы. Для явного указания исходного IP-адреса в команде `ping` следует использовать ключ `-I` с указанием IP-адреса (в данном случае 192.168.1.11):

```
# ping -c 1 -I 192.168.1.11 192.168.1.1
PING 192.168.1.1 (192.168.1.1) from 192.168.1.11 : 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=3.09 ms
--- 192.168.1.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.091/3.091/3.091/0.000 ms
```

Листинг 6.5. Указание исходного адреса в ping с помощью ключа -I

Следует отметить, что отсутствие ответа от удаленной системы на эхо-запросы не является гарантированным признаком ее неработоспособности: протокол ICMP может фильтроваться на каком-нибудь промежуточном узле или отключен на удаленном узле. В тоже время доступность удаленной системы по протоколу ICMP не означает работоспособность прикладных сетевых служб на ней: доступ к сервису

может блокироваться межсетевым экраном или сервис может быть отключен и т.д.

Трассировка маршрута с помощью команды traceroute

Для отображения всех точек маршрутизации, через которые проходят сетевые пакеты на пути следования к узлу назначения, используется команда *traceroute* (аналог в Windows - tracert). В случае проблем при доставке данных программа позволяет определить, на каком именно участке сети возникли неполадки. Для определения промежуточных маршрутизаторов *traceroute* отправляет серию (обычно три) пакетов узлу назначения, при этом каждый раз увеличивая на 1 значение поля *TTL*. Первая серия пакетов отправляется с *TTL*, равным 1, и поэтому первый маршрутизатор возвращает обратно сообщение *ICMP*, указывающее на невозможность доставки данных. Затем *traceroute* повторяет отправку серии пакетов, но уже с *TTL*, равным 2, что позволяет первому маршрутизатору пропустить их дальше. Процесс повторяется до тех пор, пока при определённом значении *TTL* пакет не достигнет узла назначения.

Для установки *traceroute* в Debian GNU\Linux следует использовать команду:

```
# apt-get install traceroute
```

Основным параметром команды *traceroute* является имя или IP-адрес удаленного узла (пример 2.1., "Сетевая подсистема Linux"). Вывод команды *traceroute* содержит информацию о точках маршрутизации (IP-адрес и/или символьное имя) и параметр *RTT* для каждой попытки серии. Ключ -n запрещает обращения к подсистеме DNS для определения символьных имен маршрутизаторов, что ускоряет работу *traceroute*.

Используя ключ -s, можно указать *traceroute*, какой исходный IP-адрес использовать (в данном случае 192.168.1.11):

```
# traceroute -s 192.168.1.11 192.168.2.5
```

Статистика интерфейсов с помощью команды netstat

Утилита `netstat` является удобным средством мониторинга сетевой активности. Фактически данная утилита служит для отображения различных структур данных, связанных с сетью.

Для отображения статистики сетевых интерфейсов следует использовать `netstat` с ключом `-i` (пример 6.6.). Все сетевые интерфейсы отображаются с помощью ключа `-a`.

```
# netstat -ai
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR T
eth0   1500 0     2012      0      0 0          244      0      0      0 BMRU
eth1   1500 0     1341      0      0 0         1118      0      0      0 BMRU
lo    16436 0        0      0      0 0            0      0      0      0 LRU
```

Листинг 6.6. Отображение статистики интерфейсов с помощью netstat

Каждая строка содержит информацию об отдельном интерфейсе: имя, MTU, метрику интерфейса (см. ниже); колонки `RX` и `TX` показывают количество пакетов, которые были получены и отправлены без ошибок (`RX-OK`, `TX-OK`), с ошибками (`RX-`*`ERR`*, `TX-`*`ERR`*), были отброшены (`RX-DRP`, `TX-DRP`), были потеряны (`RX-OVR`, `TX-OVR`). Набор символов в конце строки — флаги состояния интерфейса в стиле `ifconfig` (`B` — установлен широковещательный адрес, `L` — локальный интерфейс, `M` - "беспорядочный режим", `R` — интерфейс запущен, `U` — интерфейс поднят и т.д.)

Ключевые термины

ICMP (Internet Control Message Protocol) — сетевой протокол управляющих сообщений, используемый для диагностики стека TCP/IP.

`TTL` (Time-To-Live) — целое число, определяющее количество промежуточных маршрутизаторов, пройденных сетевым пакетом; на узле-источнике устанавливается в максимальное значение; каждый маршрутизатор отнимает от текущего значения TTL единицу.

RTT (Round Trip Time) — время между отправкой запроса и получением ответа на данный запрос.

Ping — утилита, обычно входящая в набор программного обеспечения операционной системы и позволяющая проверить доступность удаленного узла средствами протокола ICMP

Traceroute — утилита, позволяющая отобразить все промежуточные узлы на пути следования пакета до узла назначения; использует протокол ICMP.

Netstat - утилита, используемая для мониторинга сетевой активности.

Краткие итоги

Протокол ICMP (протокол сетевого уровня; использует для доставки протокол IP) предоставляет простые средства для передачи удаленным системам сообщений об исключительных ситуациях в процессе функционирования сетевого стека на узле.

Простейшим приложением протокола ICMP является утилита ping. Указав ей имя или IP-адрес удаленного узла, можно проверить его доступность.

Для трассировки маршрута (отображения всех точек маршрутизации) до удаленной системы используется ICMP -приложение *traceroute*.

Утилита netstat с ключом –i отображает статистику работы сетевых интерфейсов. Подобную статистику также содержит вывод команды ifconfig.

Упражнения

1.Изучите вывод команды ping в предложенных вариантах и определите для каждой удаленной системы ее тип (тип операционной системы):

```
ping -c 1 192.168.1.115
PING 192.168.1.115 (192.168.1.115) 56(84) bytes of data.
64 bytes from 192.168.1.115: icmp_req=1 ttl=128 time=6.66 ms
--- 192.168.1.115 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 6.664/6.664/6.664/0.000 ms

$ ping -c 1 192.168.1.103
PING 192.168.1.103 (192.168.1.103) 56(84) bytes of data.
64 bytes from 192.168.1.103: icmp_req=1 ttl=64 time=5.35 ms
--- 192.168.1.103 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.355/5.355/5.355/0.000 ms

$ ping -c 1 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 56(84) bytes of data.
64 bytes from 192.168.3.3: icmp_req=1 ttl=253 time=1.98 ms
--- 192.168.3.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.985/1.985/1.985/0.000 ms
```

2.(требуется подключение к Интернет). В среде VirtualBox запустите виртуальную машину с установленным Debian GNU\Linux с одной сетевой картой, предварительно установив тип сетевого подключения "NAT" (см. "Использование виртуализации для изучения Linux") и настройте сетевой интерфейс локальной сети с помощью файла /etc/network/interfaces на получение IP-адреса по протоколу DHCP.

2.1.Проверьте доступность узлов ссылка: http://www.mail.ru, ссылка: http://www.debian.org, ссылка: http://www.freshmeat.net

2.2.Отобразите трассу до узла ссылка: http://www.ubuntu.com

2.3.Отобразите статистику работы сетевых интерфейсов виртуальной машины.

3(требуется подключение к Интернет и наличие ОС Windows). Выполните задания 2.1.-2.3. с использованием соответствующих аналогов `ping`, `tracert` и `netstat` в Windows.

Транспортный и прикладной уровни модели сетевого взаимодействия.

Даны краткие сведения о взаимодействии систем на транспортном и прикладном уровнях модели сетевого взаимодействия. Приведены примеры использования команд ss и netstat для получения информации об активных сетевых соединениях, утилиты IPTraf для мониторинга сетевых подключений, утилиты iperf для тестирования производительности канала передачи между двумя системами. Приведен также пример использования утилиты telnet для диагностики работы протоколов прикладного уровня.

Содержание:

Основы транспортного уровня

Протоколы транспортного уровня [7] [30] реализуют разделение потоков данных между приложениями на одном компьютере, таким образом, что несколько приложений могут использовать одно и то же сетевое подключение. Разделение осуществляется с помощью портов (числовых номеров). Другими словами, для данного транспортного протокола на данном IP-адресе приложение резервирует порт для сетевого взаимодействия (другое приложение этот порт уже не может использовать). Нумерации портов различных протоколов транспортного уровня не пересекаются.

Взаимодействие между приложениями осуществляется посредством сокетов (пара IP-адрес:порт): на каждом из взаимодействующих узлов создается сокет, с которым связаны входной и выходной потоки для

передачи данных (рис 7.1).

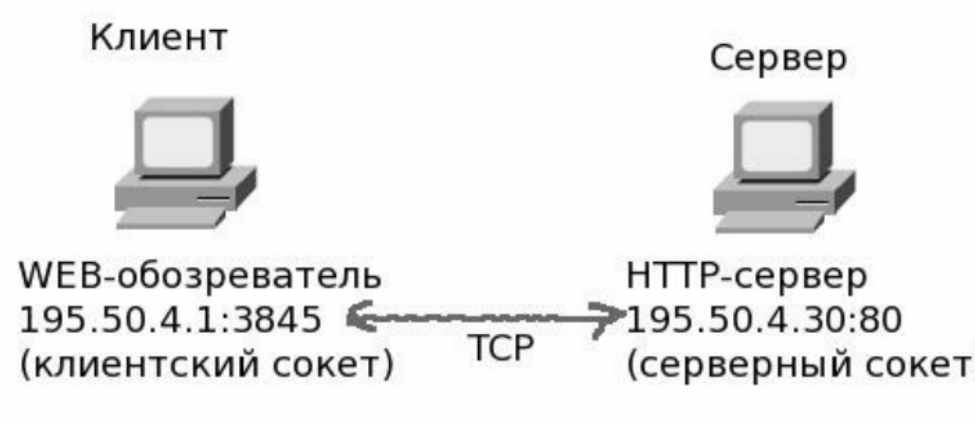

Рис. 7.1. Пример клиент-серверного взаимодействия на основе сокетов

Информация об используемых портах записывается в заголовок единицы передачи протокола транспортного уровня.

В качестве транспорта для IP используются два протокола:

1. Протокол TCP (Transmission Control Protocol) — протокол транспортного уровня с установлением соединения, управлением потоком передачи и гарантированной доставкой данных. Единицы передачи обычно называются сегментами.
2. Протокол UDP (User Datagram Protocol) — протокол транспортного уровня без установления соединения (не гарантирует доставку датаграмм — единиц передачи).

За наиболее популярными сетевыми службами номера портов закреплены стандартные значения. В Linux данная информация находится в файле /etc/services (пример 7.1).

```
$ cat /etc/services | grep "^http\|^ssh\|^www\|^pop3\|^smtp\|^ftp"
ftp-data        20/tcp
ftp     21/tcp
ssh     22/tcp          # SSH Remote Login Protocol
ssh     22/udp
smtp    25/tcp    mail
```

```
www    80/tcp    http    # WorldWideWeb HTTP
www    80/udp          # HyperText Transfer Protocol
pop3   110/tcp   pop-3   # POP version 3
pop3   110/udp  pop-3
https  443/tcp          # http protocol over TLS/SSL
https  443/udp
ftps-data  989/tcp      # FTP over SSL (data)
ftps   990/tcp
pop3s  995/tcp          # POP-3 over SSL
pop3s  995/udp
http-alt            8080/tcp  webcache  # WWW caching service
http-alt            8080/udp      # WWW caching service
```

Листинг 7.1. Фрагмент файла /etc/services

Содержимое файла /etc/services имеет рекомендательный характер, т. е. администратор системы может использовать нестандартные значения портов для сетевых служб. В Linux порты младше 1024 могут быть задействованы только суперпользователем. Для клиентских подключений используются порты со старшими номерами, выбираемые случайно из незанятых.

Максимальное количество портов, которое может использовать протокол определяется размером соответствующих полей в заголовке транспортного уровня (16 бит для TCP и UDP).

Просмотр информации о текущих соединениях и открытых портах в Linux

Для получения информации о работе транспортного уровня на узле можно использовать утилиту `ss` из пакета `iproute2` или утилиту `netstat`.

```
# ss -4antu
Netid  State      Recv-Q Send-Q     Local Address:Port       Peer Address:Port
udp    UNCONN     0      0              *:67                    *:*
tcp    LISTEN     0      128            *:22                    *:*
tcp    ESTAB      0      0      192.168.1.1:55469       192.168.1.10:22
tcp    ESTAB      0      0      192.168.56.102:22       192.168.56.1:50
```

Листинг 7.2. Использование ss для вывода информации о текущих соединениях и открытых портах

В данном примере локальный порт UDP/67 используется некоторой службой на всех IP-адресах узла (первая строка); локальный порт TCP/22 ожидает (*LISTEN*) подключений на всех IP-адресах (вторая строка); с удаленного адреса 192.168.56.1 установлено соединение (ESTAB) со службой, использующей локальный порт TCP/22 (последняя строка); с локальной системы также установлено клиентское подключение с удаленной службой на узле 192.168.1.10, использующей удаленный порт TCP/22 (третья строка).

Ключ -4 указывает выводить сведения по протоколу *IPv4*, -a — выводить всю информацию, -n -выводить узлы и порты в числовом виде; -t — протокол TCP, -u — протокол UDP. Для того, чтобы узнать какие процессы используют сетевые соединения в команде ss следует использовать ключ -p. Ключ -l отображает только слушающие порты. Более подробную информацию о команде ss можно получить на странице руководства (man ss).

```
# netstat -4antu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 192.168.1.1:55469       192.168.1.10:22         ESTABLISHED
tcp        0      0 192.168.56.102:22       192.168.56.1:50405      ESTABLISHED
udp        0      0 0.0.0.0:67              0.0.0.0:*
```

Листинг 7.3. Использование netstat для вывода информации о текущих соединениях и открытых портах

Используемые в данном примере ключи netstat аналогичны ключам ss в предыдущем примере. Более подробную информацию о команде netstat можно получить на странице руководства (man netstat).

Мониторинг сетевых соединений с помощью утилиты IPTraf

Утилита IPTraf представляет собой интерактивное средство

отображения постоянно обновляемой сетевой статистики с текстовым интерфейсом на основе библиотеки ncurses [31]. IPTraf способна отображать различные данные, включая информацию по TCP, счетчики UDP и ICMP, загрузку сети Ethernet, статистику узла и т. д.

Для установки IPTraf из стандартных репозиториев Debian можно использовать команду:

```
# apt-get install iptraf
```

Для использования IPTraf необходимы полномочия суперпользователя. IPTraf запускается в интерактивном режиме. После приветственного сообщения при нажатии любой клавиши открывается меню, в котором можно выбрать режим просмотра статистики, установить фильтры или изменить настройки программы.

Доступны следующие режимы просмотра:

1. Монитор IP-трафика (IP Traffic Monitor). Для выбранного интерфейса (всех интерфейсов) включает списки TCP -соединений и статистику по ним, а также монитор UDP -датаграмм.
2. Общая статистика интерфейсов (General interface statistics). Для каждого интерфейса отображает статистику пакетов.
3. Детальная статистика интерфейса (Detailed interface statistics). Для выбранного интерфейса отображает количество переданных и полученных единиц передачи (IP-пакетов, TCP -сегментов и UDP -датаграмм и прочих) и объем данных в байтах, а также текущую скорость передачи пакетов и байт, объемы широковещательного трафика и количество ошибок в IP-пакетах.
4. Статистический анализ (Statistical breakdowns). Для данного интерфейса отображает распределение трафика по размеру пакетов или распределение трафика по портам.
5. Монитор станции локальной сети (LAN station monitor). Для выбранного интерфейса (всех интерфейсов) показывает статистику уровня сетевых интерфейсов.

Соответствующий режим отображения может быть установлен с помощью параметров, передаваемых IPTraf при старте. Например, для

показа детальной статистики интерфейса `eth0` (рис 7.2) следует использовать ключ `-d`:

```
# iptraf -d eth0
```

Подробную информацию о ключах можно получить из руководства пользователя (`man iptraf`).

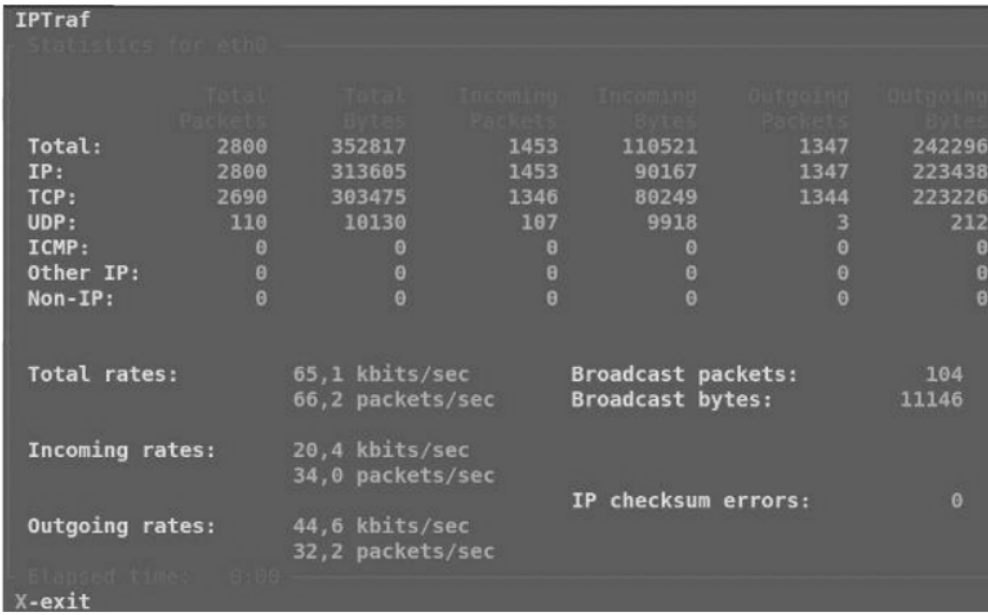

Рис. 7.2. Детальная статистика интерфейса в IPTraf

Примечание. Существует и другие программы, подобные `IPTraf` [32], [33].

Тестирование производительности сети с помощью утилиты iperf

Утилита `iperf` является простейшим средством для тестирования пропускной способности между двумя сетевыми узлами. `Iperf` представляет собой генератор `TCP/UDP` трафика и имеет клиент-серверную архитектуру [34].

Для установки данной утилиты из репозиториев Debian:

```
# apt-get install iperf
```

Для тестирования сетевого соединения на одном из узлов `iperf`,следует запустить в режиме сервера (ключ `-s`):

```
# iperf -s
```

а на другом — в режиме клиента (ключ `-c` с указанием адреса сервера):

```
$ iperf -c 192.168.1.12
```

```
$ iperf -c 192.168.1.12
------------------------------------------------------------
Client connecting to 192.168.1.12, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.3 port 51670 connected with 192.168.1.12 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  85.4 MBytes  71.6 Mbits/sec
```

Листинг 7.4. Результат работы клиента iperf

По умолчанию `iperf` использует протокол TCP. Для использования протокола UDP следует указать ключ `-u`. Более подробную информацию о параметрах `iperf` можно получить в руководстве пользователя (`man iperf`).

Примечание. Результаты работы iperf могут зависеть не только от пропускной способности канала передачи. Например, производительности узла может быть недостаточно для генерации трафика, полностью нагружающего канал; во время тестирования канал может использоваться другими сетевыми приложениями; параметры стека TCP/IP на узле могут быть не настроены оптимально и т.д.

Протоколы прикладного уровня

Прикладные протоколы обеспечивают доступ к прикладным сетевым сервисам и службам. Сетевые службы предоставляют пользователям определенные информационные услуги и обычно имеют клиент-серверную архитектуру (реже распределенную). Клиент и сервер в

рамках прикладного протокола обмениваются стандартными сообщениями (часто текстового формата) [7].

Наиболее известные прикладные протоколы:

- HTTP - протокол передачи гипертекста (основное средство передачи WEB-контента).
- Почтовые протоколы SMTP (для отправки почтовых сообщений и взаимодействия почтовых серверов), POP3 и IMAP (для доступа к почтовым ящикам).
- Протокол FTP (для обмена файлами).
- DNS — протокол взаимодействия с серверами DNS и т.д.

Многие протоколы прикладного уровня имеют защищенные версии, позволяющие шифровать передаваемую информацию между клиентом и сервером, а также идентифицировать участников сетевого взаимодействия (SSL и TLS).

Для задачи диагностики прикладных протоколов полезна утилита telnet, которая обычно присутствует в современных дистрибутивах Linux и позволяет пользователю взаимодействовать с текст-ориентированными TCP -сервисами средствами интерфейса командной строки.

Примечание. Telnet — сетевой протокол, реализующий текстовый интерфейс взаимодействия поверх TCP/IP ; ранее применялся для получения удаленного доступа к интерфейсу командной строки; используется для реализации текст-ориентированных протоколов взаимодействия.

Установка утилиты `telnet`:

```
# apt-get install telnet
```

Основными параметрами `telnet` выступают имя или адрес сервера и порт.

```
# telnet 192.168.1.10 110
Trying 192.168.1.10...
Connected to 192.168.1.10
Escape character is '^]'.
```

```
+OK test.local Zimbra POP3 server ready
user testuser
+OK hello testuser, please enter your password
pass testpassword
+OK server ready
list
+OK 6 messages
1 4763
2 7581
3 3348
4 4449
5 2437
6 3350

quit
+OK test.local closing connection
Connection closed by foreign host.
```

В данном примере с помощью утилиты `telnet` осуществляется подключение по протоколу POP3 к почтовому серверу (192.168.1.10), слушающему на 110 порту. Клиент передает команды, завершая их переводом строки (`user, pass, list, quit`), а сервер каждую команду выполняет. Прикладной протокол POP3 используется клиентом для работы с почтовым ящиком на сервере; как видно из примера является диалоговым текст-ориентированным протоколом.

Используя `telnet` можно работать с такими текст-ориентированными протоколами как HTTP, POP3, SMTP и т. д. Можно также просто проверять наличие службы на данном порту без знания непосредственно синтаксиса протокола.

Ключевые термины

Порт — номер, используемый для идентификации канала передачи приложения, работающего по данному протоколу транспортного уровня.

Сокет — оконечная точка канала связи, с помощью которой приложение может использовать сетевое подключение (пара IP-адрес:порт

однозначно определяющая сетевой процесс).

TCP (Transmission Control Protocol) — протокол транспортного уровня с установлением соединения, управлением потоком передачи и гарантированной доставкой данных.

UDP (User Datagram Protocol) — протокол транспортного уровня без установления соединения и не гарантирующий доставку.

Сегменты — единицы передачи протокола TCP.

Датаграммы — единицы передачи протокола UDP.

Клиент-серверная архитектура — подход к организации информационной системы, при котором сетевая нагрузка распределены между сервером (поставщиком услуг) клиентом (потребителем услуг).

HTTP (HyperText Transfer Protocol) - прикладной протокол передачи WEB-контента.

SMTP (Simple Mail Transfer Protocol) — прикладной почтовый протокол, используемый для отправки сообщений электронной почты и взаимодействия серверов электронной почты друг с другом.

POP3 (Post Office Protocol Version 3) — прикладной почтовый протокол, используемый программой-клиентом электронной почты для получения корреспонденции с сервера.

IMAP (Internet Message Access Protocol) — протокол прикладного уровня для доступа к электронной почте с расширенными возможностями.

FTP (File Transfer Protocol) — прикладной протокол, используемый для передачи файлов

SSL (Secure Sockets Layer) - криптографический протокол обеспечивающий защищенное соединение между клиентом и сервером.

TLS (Transport Layer Security) — открытая версия протокола SSL.

Краткие итоги

1. Протоколы транспортного уровня реализуют разделение потоков данных между приложениями на одном компьютере с помощью портов. Нумерации портов различных протоколов транспортного уровня не пересекаются. Номера портов записываются в заголовок единицы передачи соответствующего транспортного протокола.
2. Поверх IP используются два транспортных протокола: TCP (для надежной доставки) и UDP (для " быстрой" доставки).
3. Утилиты `ss` и `netstat` позволяют получить различную информацию об активных сетевых соединениях и открытых портах.
4. Утилита `IPTraf` позволяет в " режиме реального времени" получать информацию о работе сети и взаимодействии систем друг с другом.
5. Для тестирования полосы пропускания канала связи между двумя системами можно использовать утилиту `iperf`.
6. Большинство протоколов прикладного уровня являются текст-ориентированными (взаимодействующие системы обмениваются текстовыми сообщениями друг с другом).

Упражнения

1. (требуется Windows -совместимая ОС). Используя утилиту `netstat` Windows определите, какие TCP и UDP -порты открыты на Вашем компьютере.
2. В среде VirtualBox запустите две виртуальные машины с установленным Debian GNU\Linux с одной сетевой картой у каждой, настроенной на тип сетевого подключения "Внутренняя сеть". Установите на виртуальных машинах IP-адреса из одной подсети. Используя утилиту `iperf`, протестируйте пропускную способность виртуальной сети VirtualBox.
3. Некоторая сетевая TCP -служба не запускается в системе с Linux.

Укажите более вероятную причину:

1. Описание порта, который использует служба, отсутствует в файле /etc/services.
2. Порт занят другим приложением.

Настройка некоторых сетевых служб в Debian GNU/Linux

Приведены простые примеры настройки серверов ssh, apache и squid.

Содержание:

Сервер защищенных соединений (ssh)

Протокол SSH (Secure SHell) позволяет получить удаленный доступ к компьютеру по безопасному соединению (работает на уровне потоков данных) [35], [28]. Существуют две версии этого протокола: SSH-1 (считается ненадежной) и SSH-2 (рекомендована к использованию). SSH реализуется двумя приложениями: сервером и клиентом. При подключении клиент проходит процедуру аутентификации и между клиентом и сервером устанавливается защищенное соединение.

В Linux для реализации SSH используется программное обеспечение OpenSSH [36]. Для установки сервера в Debian можно использовать команду:

```
# apt-get install openssh-server
```

В процессе установки сервер автоматически прописывается в автозагрузку и запускается. По умолчанию сервер слушает на TCP-порту с номером 22 на всех интерфейсах:

```
# ss -4nlp sport = :ssh
Recv-Q Send-Q        Local Address:Port      Peer Address:Port
0      128               *:22            *:*         users:(("sshd",983,3))
```

Для остановки/запуска/перезапуска сервера можно использовать скрипт `ssh` в каталоге /etc/init.d с параметром `stop/start/`*`restart`* соответственно, например для перезапуска:

```
# /etc/init.d/ssh restart
```

Конфигурационный файл сервера sshd_config расположен в каталоге /etc/ssh. В настройках по умолчанию указано использовать аутентификацию на основе публичных ключей (`PubkeyAuthentication yes`) или логина и пароля (`PasswordAuthentication yes`), разрешать удаленное подключение суперпользователя (`PermitRootLogin yes`), использовать SSH-2 (`Protocol 2`), не разрешать пустые пароли (`PermitEmptyPasswords no`) и т. д. Для применения внесенных изменений в конфигурацию необходимо перезапустить сервер.

Для того, чтобы сервер `ssh` слушал на адресе `192.168.56.102` необходимо в файле /etc/ssh/sshd_config раскомментировать и исправить опцию `ListenAddress`:

```
ListenAddress 192.168.56.102
```

Листинг 8.1. Изменение настроек сервера ssh

Затем перезапустить openssh-server:

```
# /etc/init.d/ssh restart
```

Результат можно проверить командой `ss`:

```
# ss -4nl sport = :ssh
Recv-Q Send-Q      Local Address:Port         Peer Address:Port
  0    128       192.168.56.102:22                *:*
```

Для получения удаленного доступа к командному интерпретатору на клиентской системе необходимо использовать утилиту `ssh`, которая обычно присутствует в базовой установке системы (ее также можно установить, используя команду *`apt`*`-get install openssh-client`). В качестве параметра надо указать `имя пользователя` и `IP-адрес (имя удаленной системы)` используя символ `@` в качестве разделителя (пример 8.2):

```
$ ssh root@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
```

```
RSA key fingerprint is d7:33:4d:e3:0b:e2:ec:57:48:34:f1:77:88:ab:77:de.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.102' (RSA) to the list of known hosts.
root@192.168.56.102's password:
Linux ssl1 2.6.32-5-686 #1 SMP Wed Jan 12 04:01:41 UTC 2011 i686
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the exten
permitted by applicable law.
Last login: Wed Apr  6 13:19:19 2011 from 192.168.56.1
root@ssl1:~#
```

Листинг 8.2. Первое подключение к удаленной системе

При первом подключение будет выдано предупреждение о неизвестном узле. При ответе `yes`, удаленный узел будет добавлен в список известных узлов (файл ~/.ssh/known_hosts) и при последующих подключениях предупреждение не будет выдаваться. Затем нужно ввести пароль. После успешного ввода пароля будет предоставлен доступ к интерфейсу командной строки удаленной системы. Для завершения удаленного сеанса следует использовать команду `exit`.

Для копирования файлов по протоколу ssh можно использовать утилиту scp, при этом на одной из систем должен быть установлен ssh-сервер (примеры 8.3 и 8.4).

```
# scp root@192.168.56.102:/boot/initrd.img-2.6.32-5-686 .
root@192.168.56.102's password:
initrd.img-2.6.32-5-686                 100% 8318KB   8.1MB/s   00:01
```

Листинг 8.3. Копирование файла из удаленной системы

В данном примере файл копируется из указанного пути (/boot/initrd.img-2.6.32-5-686) удаленной системы (192.168.56.102) в текущий каталог локальной системы (.). Для этого используется учетная запись суперпользователя удаленной системы.

```
$ scp known_hosts root@192.168.56.102:/root
```

Листинг 8.4. Копирование файла на удаленную систему

В данном примере файл (known_hosts) копируется из текущего каталога локальной системы в каталог суперпользователя (/root) удаленной системы.

Для копирования каталогов в команде *scp* нужно использовать ключ *-r*.

Примечание: Для операционных систем семейства Windows также существуют программы клиенты ssh [37], [38].

HTTP-сервер Apache

HTTP-сервер Apache является программным обеспечением с открытым исходным кодом для операционных систем Linux, Unix, Windows, Mac и Netware. Главной целью проекта является обеспечение защищенных, эффективных и расширяемых HTTP-сервисов, поддерживающих современные стандарты [39].

Для установки Apache в Debian можно использовать команду:

```
# apt-get install apache2
```

В процессе установки сервер прописывается в автозагрузку и запускается. По умолчанию сервер слушает на всех интерфейсах на TCP-порту с номером 80. Для остановки/запуска/перезапуска сервера можно использовать скрипт *apache2* в каталоге /etc/init.d с параметром *stop/start/restart* соответственно:

```
# /etc/init.d/apache2 restart
```

Конфигурационные файлы Apache расположены в каталоге /etc/apache2. В частности настройки сайта по умолчанию расположены в файле /etc/apache2/sites-available/default:

```
# cat /etc/apache2/sites-available/default
 <VirtualHost *:80>
   ServerAdmin webmaster@localhost
       DocumentRoot /var/www
        <Directory />
```

```
            Options FollowSymLinks
            AllowOverride None
        </Directory>
        <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride None
            Order allow,deny
            allow from all
        </Directory>

        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
            AllowOverride None
            Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
            Order allow,deny
            Allow from all
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        LogLevel warn
        CustomLog ${APACHE_LOG_DIR}/access.log combined

    Alias /doc/  "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>

</VirtualHost>
```

В конфигурационных файлах Apache используется теговый подход. В данном случае в теге виртуального узла (`VirtualHost`) описаны его настройки. Присутствуют несколько "дочерних" тегов с описанием каталогов, доступных данному виртуальному узлу (Directory). Между тегами `<Directory>` для каталога можно указазать различные опции (директивы). Например. для каталога /var/www установлен приоритет

разрешительных директив (`Order allow, deny`), описывающих правила доступа к данному каталогу; также разрешен доступ всем (`allow from all`) и т.д. Основным каталогом для документов является /var/www (директива DocumentRoot). Директива Alias описывает виртуальный путь HTTP-сервера, т.е. все запросы вида `http://<IP-адрес сервера> /doc/` будут перенаправляться в каталог /usr/share/doc.

Для проверки работоспособности WEB-сервера можно используя Интернет-обозреватель, указав в строке адреса IP-адрес HTTP-сервера. Например (в данном случае используется текстовый браузер с указанием открываемой страницы в качестве параметра):

```
# lynx http://192.168.56.102
```

Протоколы работы сайта по умолчанию расположены в каталоге /var/log/apache2.

Apache имеет широкий набор функциональных возможностей, большое количество различных настроек и может использоваться в различных схемах WEB-доступа. Более подробную информацию по настройке и применению Apache можно найти в официальной документации [41], книгах и пособиях, посвященных этой программе, а также в сети Интернет.

Кэширующий прокси-сервер

Кэширующий прокси-сервер Squid [42] представляет собой реализацию прокси прикладного уровня для WEB-трафика (протоколы HTTP, HTTPS, FTP и т.д.). Обычно squid устанавливается на компьютере, имеющем как минимум два сетевых интерфейса: один с доступом в сети Интернет, а другой — в локальную сеть. Интернет-обозреватель или программа, взаимодействующая с прокси, настраивается на адрес прокси в локальной сети и заданный порт и далее все WEB-запросы отправляет прокси-серверу. Прокси обслуживает запросы клиентов, используя Интернет-интерфейс. При этом часть WEB-трафика сохраняется на файловой системе прокси (кэшируется), а при повторных запросах клиентов данные извлекаются из кэша.

Пример 8.5. Пример простой настройки Squid в Debian

Используя VirtualBox, создадим простой стенд для отладки схемы доступа в Интернет на основе Squid. Стенд будет включать две виртуальные машины на основе Debian: прокси-сервер и клиент локальной сети.

На прокси-сервере:

1.В VirtualBox предварительно для виртуальной машины включим два сетевых интерфейса: один в сеть NAT — Интернет-интерфейс (полагаем, что основная система имеет доступ в Интернет), а второй — во внутреннюю сеть (локальный интерфейс)

2.Включим виртуальную машину и настроим интерфейсы: eth0 — Интернет-интерфейс, eth1 — интерфейс локальной сети (фрагмент /etc/network/interfaces):

```
auto eth0
iface eth0 inet dhcp

auto eth1
iface eth1 inet static
    address 192.168.1.1
    netmask 255.255.255.0
```

Перезапустим сетевую подсистему:

```
# /etc/init.d/networking restart
```

В результате интерфейс eth0 получит настройки по протоколу DHCP (от NAT-сети VirtualBox):

```
# ip addr show eth0
2: eth0:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu 1500 qdisc pfif
    link/ether 08:00:27:0a:41:d4 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global eth0
    inet6 fe80::a00:27ff:fe0a:41d4/64 scope link
       valid_lft forever preferred_lft forever
```

Также будут автоматически прописаны настройки DNS-сервера (такие как в хостовой системе):

```
# cat /etc/resolv.conf
nameserver 10.10.10.10
```

3.Установим squid:

```
#apt-get install squid
```

В процессе установки сервер прописывается в автозагрузку, создает структуру каталогов для дискового кэша (/var/spool/squid) и запускается.

4.Конфигурационный файл Squid - /etc/squid/squid.conf. По умолчанию удаленный доступ к squid запрещен. Для разрешения доступа нужно в конфигурационный файл добавить разрешающее правило для сети `192.168.1.0/24`. Для этого в секции ACCESS CONTROLS впишем список доступа (access list):

```
acl mynet src 192.168.1.0/24
```

а затем в секции http_access после правила `http_access allow localhost` добавим:

```
http_access allow mynet
```

Также исправим опцию http_port, что squid принимал клиентские запросы только на локальном интерфейсе:

```
http_port 192.168.1.1:3128
```

Перезапустим squid:

```
# /etc/init.d/squid restart
```

5.Запустим виртуальную машину клиента, предварительно установив в VirtualBox сетевой адаптер в режим внутренней сети. В системе установим IP-адрес из сети `192.168.1.0/24`. Проверим работу прокси-сервера с помощью текстового браузера lynx (в lynx опцию прокси-сервера можно установить с помощью переменной окружения):

```
# env http_proxy="http://192.168.1.1:3128"  lynx http://intuit.ru
```

Примечание: Протоколы работы squid хранятся в каталоге /var/log/squid. Файл access.log — протокол запросов, cache.log - общий лог, store.log — лог кэша.

Программное обеспечение squid имеет широкие функциональные возможности и может работать в различных режимах. Более подробную информацию о настройке и использовании squid можно получить из официальной документации, книг и ресурсов сети Интернет.

Упражнения

1. Воспроизвести с использованием VirtualBox пример 8.1, пример 8.2, пример 8.3, пример 8.4. В качестве клиента можно воспользоваться еще одной виртуальной машиной или хостовой системой (в случае Windows использовать средства ssh, доступные для данной ОС).
2. В VirtualBox воспроизвести пример 8.5.

Анализ сетевого трафика как метод диагностики сети

Даны краткие сведения об анализе сетевого трафика и консольной утилите tcpdump. Приведены примеры анализа трафика протоколов ARP, IP, TCP и UDP

Содержание

"Прослушивание" сетевого трафика

В некоторых случаях для обнаружения проблем функционирования сетевого стека узла и сегментов сети используется анализ сетевого трафика. Существуют средства, которые позволяют отобразить (прослушать) и проанализировать работу сети на уровне передаваемых фреймов, сетевых пакетов, сетевых соединений, датаграмм и прикладных протоколов.

В зависимости от ситуации для диагностики может быть доступен как трафик узла, на котором производится прослушивание сетевого трафика, так и трафик сетевого сегмента, порта маршрутизатора и т. д. Расширенные возможности для перехвата трафика основаны на "беспорядочном" (promiscuous) режиме работы сетевого адаптера: обрабатываются все фреймы (а не только те, которые предназначены данному MAC-адресу и широковещательные, как в нормальном режиме функционирования).

В сети Ethernet существуют следующие основные возможности прослушивания трафика:

- В сети на основе концентраторов весь трафик домена коллизий доступен любой сетевой станции.
- В сетях на основе коммутаторов сетевой станции доступен ее

трафик, а также весь широковещательный трафик данного сегмента.

- Некоторые управляемые коммутаторы имеют функцию копирования трафика данного порта на порт мониторинга ("зеркалирование",мониторинг порта).
- Использование специальных средств (ответвителей), включаемых в разрыв сетевого подключения и передающих трафик подключения на отдельный порт.
- "Трюк" с концентратором — порт коммутатора, трафик которого необходимо прослушать, включают через концентратор, подключив к концентратору также узел-монитор (при этом в большинстве случаев уменьшается производительность сетевого подключения).

Существуют программы (сетевые мониторы или анализаторы, sniffer), которые реализуют функцию прослушивания сетевого трафика (в т.ч. в беспорядочном режиме), отображения его или записи в файл. Дополнительно ПО для анализа может фильтровать трафик на основе правил, декодировать (расшифровать) протоколы, считать статистику и диагностировать некоторые проблемы.

Примечание: Хорошим выбором базового инструмента для анализа сетевого трафика в графической среде является бесплатный пакет wireshark [43], доступный для Windows и в репозиториях некоторых дистрибутивов Linux.

Утилита tcpdump

Консольная утилита `tcpdump` входит в состав большинства Unix-систем и позволяет перехватывать и отображать сетевой трафик [44]. Утилита использует `libpcap`, переносимую C/C++ библиотеку для перехвата сетевого трафика.

Для установки `tcpdump` в Debian можно использовать команду:

```
# apt-get install tcpdump
```

Для запуска данной утилиты необходимо иметь права суперпользователя (в частности, в связи с необходимостью перевода

сетевого адаптера в "беспорядочный" режим). В общем виде формат команды имеет следующий вид:

```
tcpdump <опции> <фильтр-выражение>
```

Для вывода на консоль описание заголовков (расшифрованные данные) перехваченных пакетов необходимо указать интерфейс для анализа трафика (опция `-i`):

```
# tcpdump -i eth0
```

Можно отключить преобразования IP адресов в доменные имена (т.к. при больших объемах трафика создается большое число запросов к DNS-серверу) — опция `-n`:

```
# tcpdump -n -i eth0
```

Для вывода данных канального уровня (например, mac адреса и прочее) - опция `-e`:

```
# tcpdump -en -i eth0
```

Вывод дополнительной информации (например, TTL, опции IP) — опция `-v`:

```
# tcpdump -ven -i eth0
```

Увеличение размера захватываемых пакетов (больше 68 байт по умолчанию) — опция `-s` с указанием размера (`-s 0` — захватывать пакеты целиком):

```
# tcpdump -s 0
```

Запись в файл (непосредственно пакеты - "дамп") - опция `-w` с указанием имени файла:

```
# tcpdump -w traf.dump
```

Чтение пакетов из файла — опция `-r` с указанием имени файла:

```
# tcpdump -r traf.dump
```

По умолчанию `tcpdump` работает в беспорядочном режиме. Ключ `-p` указывает `tcpdump` перехватывать только трафик, предназначенный данному узлу.

Дополнительную информацию по ключам и формате фильтров `tcpdump` можно получить в справочном руководстве (`man tcpdump`).

Анализ трафика на уровне сетевых интерфейсов и сетевом уровне с помощью tcpdump

Для выделения Ethernet-фреймов используются следующие конструкции `tcpdump` (общий вид):

```
tcpdump ether { src | dst | host } MAC_ADDRESS
```

где `src` — MAC-адрес источника, *`dst`* — MAC-адрес назначения, `host` — `src` или *`dst`*, а также для выделения широковещательного трафика:

```
tcpdump ether  broadcast
```

Например:

```
# tcpdump -n -i vlan0 ether src 0:2:b3:d8:d8:2c
# tcpdump -n -e -i vlan0 ether broadcast
```

В первом случае выбираются с интерфейса vlan0 фреймы с указанным MAC-адресом источника. Во втором - выбирается широковещательный трафик на интерфейсе vlan0.

Фильтрация по IP адресам (`net` — сеть, для указания маски подсети - `mask`):

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

Например:

```
# tcpdump -n -i eth0 src 192.168.66.1
```

```
# tcpdump -n -i eth0 host 192.168.66.1
# tcpdump -n -i eth0 src net 10.0.0.0 mask 255.0.0.0
```

В первом случае фильтруются сетевые пакеты, в заголовке которых в поле источник указан IP-адрес 192.168.66.1. Во втором случае — пакеты, в которых данный IP-адрес указан как источник или как получатель пакета. В третьем — пакеты, в которых источником указаны узлы сети 10.0.0.0/8.

Фильтрация по IP протоколу:

```
tcpdump { arp | rarp | ip | tcp | udp | icmp }
```

Например, выбирать ICMP-пакеты:

```
# tcpdump -n -i eth0 icmp
```

Сложные фильтры могут содержать множество примитивов, связанных между собой с использованием логических операторов `and`, `or` и `not`.

Например: `host 192.168.12.5 and host 192.168.13.4`.

Анализ работы протокола ARP

Используем хостовую систему и виртуальную машину с сетевым адаптером, настроенным на виртуальную сеть узла. Установим на сетевом адаптере виртуальной машины IP-адрес из той же подсети, что и на адаптер vboxnet0 (адаптер виртуальной сети узла) основной системы.

Настройки адаптера vboxnet0 в основной системе:

```
# ip addr show dev vboxnet0
5: vboxnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdis
   state UNKNOWN qlen 1000
   link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
   inet 192.168.56.1/24 brd 192.168.56.255 scope global vboxnet0
   inet6 fe80::800:27ff:fe00:0/64 scope link
```

```
    valid_lft forever preferred_lft forever
```

Настройки сетевого адаптера в виртуальной машине:

```
# ip addr show dev eth1
3: eth1:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu 1500 qdisc
   pfifo_fast state UP qlen 1000
   link/ether 08:00:27:fd:e5:aa brd ff:ff:ff:ff:ff:ff
   inet 192.168.56.33/24 scope global eth1
   inet6 fe80::a00:27ff:fefd:e5aa/64 scope link
     valid_lft forever preferred_lft forever
```

Можно проверить командой `ip neigh show dev vboxnet0`, что в ARP-таблице основной системы нет записей связанных с интерфейсом vboxnet0.

Откроем еще одну консоль на основной системе и запустим `tcpdump`:

```
# tcpdump -ne -i vboxnet0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vboxnet0, link-type EN10MB (Ethernet), capture size 65535 bytes
```

Запустим `ping` виртуальной машины с основной системы:

```
# ping -c 1 192.168.56.33
 PING 192.168.56.33 (192.168.56.33) 56(84) bytes of data.
 64 bytes from 192.168.56.33: icmp_req=1 ttl=64 time=1.06 ms
 --- 192.168.56.33 ping statistics ---
 1 packets transmitted, 1 received, 0% packet loss, time 0ms
 rtt min/avg/max/mdev = 1.067/1.067/1.067/0.000 ms
```

В ARP-таблице основной системы появилась запись об IP-адресе виртуальной машины:

```
# ip neigh show dev vboxnet0
192.168.56.33 lladdr 08:00:27:fd:e5:aa STALE
```

Вывод `tcpdump`:

```
# tcpdump -ne -i vboxnet0
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vboxnet0, link-type EN10MB (Ethernet), capture size 65535 bytes
12:16:29.465780 0a:00:27:00:00:00 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806),
    length 42: Request who-has 192.168.56.33 tell 192.168.56.1, length 28
12:16:29.466786 08:00:27:fd:e5:aa > 0a:00:27:00:00:00, ethertype ARP (0x080
    length 42: Reply 192.168.56.33 is-at 08:00:27:fd:e5:aa, length 28
12:16:29.466815 0a:00:27:00:00:00 > 08:00:27:fd:e5:aa, ethertype IPv4 (0x080
  length 98: 192.168.56.1 > 192.168.56.33: ICMP echo request, id 5284, seq 1
12:16:29.467934 08:00:27:fd:e5:aa > 0a:00:27:00:00:00, ethertype IPv4 (0x080
  length 98: 192.168.56.33 > 192.168.56.1: ICMP echo reply, id 5284, seq 1, l
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
```

Каждая запись о сетевом пакете в таком формате содержит время перехвата пакета, MAC-адреса отправителя и получателя, тип протокола, длину пакета и сведения о содержимом пакета. Первая запись описывает широковещательный ARP-запрос с MAC-адреса интерфейса vboxnet0 основной системы ("У кого адрес 192.168.56.33 ответь 192.168.56.1"). Вторая запись — ответ с MAC-адреса виртуальной машины на MAC-адрес основной системы ("192.168.56.33 имеет определенный MAC-адрес"). Третья и четвертая записи (ICMP-запрос и ICMP-ответ) являются результатом работы команды ping на основной системе.

Таким образом, основная система для того, чтобы отправить стандартный эхо-запрос виртуальной машине, предварительно по протоколу ARP получила MAC-адреса виртуальной машины и внесла его с привязкой к IP-адресу в свою ARP-таблицу.

Работа `tcpdump` была прервана комбинацией клавиш Ctrl+C. Перед завершением работы `tcpdump` печатает статистику работы: количество перехваченных, полученных фильтром и отброшенных ядром пакетов.

Анализ трафика на транспортном уровне с помощью tcpdump

Для фильтрации трафика определенного порта на транспортном уровне можно использовать ключевое слово `port`. Например:

```
# tcpdump -n -i eth0 src port 53
# tcpdump -n -i eth0 tcp port 80
```

В первом случае будут отбираться пакеты, несущие единицы передачи транспортного уровня (TCP или UDP), в которых порт-источник установлен в `53`. Во втором случае — пакеты с TCP-сегментами, у которых порт источника или назначения равен `80`.

Обратимся с основной системы к WEB-серверу, установленному на виртуальной машине:

```
# lynx 192.168.56.33
```

предварительно, запустив `tcpdump` на другой консоли:

```
# tcpdump -n -i vboxnet0 host 192.168.56.33 and port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vboxnet0, link-type EN10MB (Ethernet), capture size 65535 bytes
15:44:37.837393 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [S], seq 1
    win 5840, options [mss 1460,sackOK,TS val 2934847 ecr 0,nop,wscale 7],
15:44:37.838118 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [S.], seq 3
    ack 1209026236, win 5792, options [mss 1460,sackOK,TS val 3275261 e
    2934847,nop,wscale 2], length 0
15:44:37.838157 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [.], ack 1,
    win 46, options [nop,nop,TS val 2934847 ecr 3275261], length 0
15:44:37.839254 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [P.], seq 1
    ack 1, win 46, options [nop,nop,TS val 2934847 ecr 3275261], length 221
15:44:37.839921 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [.], ack 2
    win 1716, options [nop,nop,TS val 3275262 ecr 2934847], length 0
15:44:37.848118 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [P.], seq 1
    ack 222, win 1716, options [nop,nop,TS val 3275264 ecr 2934847], length
15:44:37.848156 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [.], ack 4
    win 54, options [nop,nop,TS val 2934848 ecr 3275264], length 0
15:44:37.849738 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [F.], seq 4
    ack 222, win 1716, options [nop,nop,TS val 3275264 ecr 2934848], length
15:44:37.850366 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [F.], seq 2
    ack 447, win 54, options [nop,nop,TS val 2934848 ecr 3275264], length 0
```

```
15:44:37.851267 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [.], ack 2
    options [nop,nop,TS val 3275265 ecr 2934848], length 0
…
```

Листинг 9.2. Анализ работы протокола TCP

В данном примере клиент (192.168.56.1) с TCP-порта 41533 устанавливает соединение с сервером (192.168.56.33), слушающим на порту 80, делает запрос, получает необходимые данные и соединение завершается.

Заголовок TCP-сегмента, помимо номеров портов получателя и отправителя содержит ряд параметров [7]:

- Номер последовательности (`seq`). Определяет порядок следования байт в отправляемом в сеть потоке (смещение первого байта в сегменте относительно начала потока данных).
- Подтвержденный номер (*ACK*). Максимальный номер байта в полученном сегменте увеличенный на 1. Отправляемые отправителю подтверждения одновременно служат запросом новой порции данных.
- Управляющие флаги (*SYN* - S, *ACK*, FIN -F , *RST* – R , PSH - P, URG)
- Окно (`win`) — количество байтов данных, ожидаемых отправителем данного, начиная с байта номер которого указан в поле *ack*. Для оптимизации передачи отправитель не ждет подтверждения для каждого отправленного сегмента, а может отправить в сеть группу сегменту (но в байтах не больше размера окна). Если качество канала плохое (много запросов на повторную передачу, теряются подтверждения) окно уменьшается, если хорошее — окно увеличивается.
- Опции. Используются для решения вспомогательных задач. Например, передается MSS (Maximum segment size) — максимальный размер сегмента.

Процесс установления двунаправленного соединения по протоколу TCP отражают первые три записи `tcpdump`:

- Клиент отправляет серверу TCP-сегмент с установленым флагом

SYN, начальным "случайным" номером (`1209026235`), с которого будут нумероваться байты в отправляемом им потоке, максимальный размер окна — объем, который разрешено передавать серверу без подтверждения от клиента (5840):

```
15:44:37.837393 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [S]
win 5840, options [mss 1460,sackOK,TS val 2934847 ecr 0,nop,wsca
```

- Сервер отправляет клиенту TCP-сегмент с установленными флагами *SYN* и *ACK*, начальным "случайным" номером (`370041518`), с которого будут нумероваться байты в отправляемом им потоке, и максимальный размер окна для клиента (5792). Данный сегмент также является подтверждением получения запроса на установление соединения от клиента:

```
15:44:37.838118 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [S.
ack 1209026236, win 5792, options [mss 1460,sackOK,TS val 32752
```

- Клиент отправляет серверу TCP-сегмент с установленным флагом *ACK*, который является подтверждением получения сегмента от сервера (далее `tcpdump` отображает относительные значения `seq` и `ask`):

```
15:44:37.838157 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [.],
options [nop,nop,TS val 2934847 ecr 3275261], length
```

После этого соединение считается установленным.

В следующей паре записей клиент передает в секции данных сегмента серверу запрос протокола прикладного уровня (221 байт) и получает от сервера подтверждение его получения:

```
15:44:37.839254 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [P.], seq 1
ack 1, win 46, options [nop,nop,TS val 2934847 ecr 3275261], length 221
15:44:37.839921 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [.], ack 2
options [nop,nop,TS val 3275262 ecr 2934847], length 0
```

При этом флаг `PSH (P)` используется для оповещения отправляющей стороны о том, что принимающая готова принимать данные.

Далее сервер отправляет данные клиенту (445 байт) и получает от него подтверждение получения:

```
15:44:37.848118 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [P.], seq 1
  ack 222, win 1716, options [nop,nop,TS val 3275264 ecr 2934847], length 4
15:44:37.848156 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [.], ack 4
  win 54, options [nop,nop,TS val 2934848 ecr 3275264], length 0
```

Затем по инициативе сервера соединение завершается. Сервер шлет сегмент с установленным флагом `FIN`:

```
15:44:37.849738 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [F.], seq 4
  ack 222, win 1716, options [nop,nop,TS val 3275264 ecr 2934848], length 0
```

Клиент в ответ также отправляет сегмент с установленным флагом `FIN` ; данный сегмент одновременно является подтверждением получения запроса на завершение соединения от сервера:

```
15:44:37.850366 IP 192.168.56.1.41533 > 192.168.56.33.80: Flags [F.], seq 2
  ack 447, win 54, options [nop,nop,TS val 2934848 ecr 3275264], length 0
```

Серверу остается лишь подтвердить получение `FIN-сегмента` от клиента.

```
15:44:37.851267 IP 192.168.56.33.80 > 192.168.56.1.41533: Flags [.], ack 2
  win 1716, options [nop,nop,TS val 3275265 ecr 2934848], length 0
```

```
21:56:14.381091 IP 192.168.56.1.54040 > 192.168.56.33.23: Flags [S], seq 2
  win 5840, options [mss 1460,sackOK,TS val 5164501 ecr 0,nop,wscale 7], l
21:56:14.381688 IP 192.168.56.33.23 > 192.168.56.1.54040: Flags [R.],
  seq 0, ack 2956835312, win 0, length 0
```

Листинг 9.3. Реакция на попытку подключения к закрытому порту TCP

В данном примере с системы 192.168.56.1 делается попытка подключится к несуществующему TCP-сервису на узле 192.168.56.33. Удаленная система реагирует отправкой сегмента с установленным флагом *`RST`* (сброса соединения).

```
21:55:16.925906 IP 192.168.56.1.41979 > 192.168.56.33.53: 6561+ A? www
```

```
21:55:16.926615 IP 192.168.56.33 > 192.168.56.1: ICMP 192.168.56.33 udp
```

Листинг 9.4. Реакция на попытку отправки UDP-датаграммы на закрытый порт

В данном примере сделана попытка отправить UDP-датаграмму на несуществующий порт удаленной системы. Протокол UDP обычно реагирует отправкой ICMP-сообщения исходному узлу о недостижимости порта.

Ключевые термины

Беспорядочный (promiscuous) режим — режим работы сетевого адаптера, при котором принимаются все фреймы, а не те, которые предназначены данному узлу, как в обычном состоянии.

Сетевой монитор или анализатор (sniffer) — средство для перехвата и анализа сетевого трафика.

Краткие итоги

1. Для анализа трафика узла (а также для анализа трафика сетевого сегмента) можно использовать специальные средства — сетевые анализаторы или мониторы.
2. В Unix-подобных ОС утилита tcpdump позволяет выполнять базовые операции по перехвату сетевого трафика, декодирования протоколов, записи и чтения сетевого трафика в файл, фильтрации трафика для анализа и т. д.
3. Для анализа трафика в графическом режиме пользователя можно использовать пакет wireshark.

Упражнения

1.В среде VirtualBox запустите виртуальную машину (ВМ) с установленным Debian GNU\Linux и установленным HTTP-сервером с одной сетевой картой, предварительно установив тип сетевого

подключения "Виртуальная сеть узла" .

Изучите процесс взаимодействия между HTTP-сервером, установленным на ВМ, и клиентом на основной системе:

1.1.Запустите на консоле ВМ утилиту `tcpdump` с параметрами `-nve`.

1.2.В основной системе запустите командный интерпретатор и используя утилиту `telnet` установите TCP соединение с HTTP-сервером, слушающим на 80 порту ВМ:

```
# telnet <IP-адрес ВМ> 80
```

Затем отправьте ВМ запрос корневой страницы, закончив ввод двойным нажатием клавиши enter:

```
GET / HTTP/1.1
Host: <IP-адрес ВМ>
```

Сервер должен выдать ответ с кодом `200 (HTTP/1.1 200 OK)`

1.3.Переключитесь в ВМ и, используя результаты работы `tcpdump`, последовательно изучите ход сетевого соединения.

2.Для данного упражнения необходимо, чтобы в виртуальной машине с установленным Debian GNU\Linux были установлены HTTP и SSH-сервера. Сетевая карта виртуальной машины должна быть подключена к виртуальной сети узла VirtualBox.

2.1.Запустите на консоли виртуальной машины `tcpdump` таким образом, чтобы перехваченные пакеты в низкоуровневом формате записывались в файл (дамп трафика). Далее:

- Проверьте доступность виртуальной машины с основной с помощью `ping`.
- Проверьте доступность основной системы с виртуальной с помощью `ping`.
- Запустите Интернет-обозреватель в основной системе и запросите страницу с HTTP-сервера виртуальной машины.
- Затем, используя клиент `ssh` (для Windows можно использовать

putty или winscp), подключитесь с основной системы к виртуальной машине и отключитесь).

- Используя утилиту telnet, попробуйте подключиться с основной системы к виртуальной машине на порт 8080.
- Попробуйте с виртуальной машины сделать DNS-запрос на разрешение имени ссылка: http://www.yandex.ru, используя утилиту nslookup (*apt*-get install dnsutils).
- Удалите из ARP-таблицы виртуальной машины запись о MAC-адресе основной системы, а затем проверьте доступность основной системы с помощью ping.

Затем прервите работу tcpdump комбинацией клавиш Ctrl+C.

2.2.Сконструируйте правила tcpdump, которые бы из полученного файла дампа отображали:

2.2.1.Все ICMP-пакеты

2.2.2.ICMP-пакеты, отправленные основной системой

2.2.3.Взаимодействие по протоколу TCP между основной системой и виртуальной машиной.

2.2.4.Взаимодействие между HTTP-клиентом и HTTP-сервером.

2.2.5.Взаимодействие между основной системой и TCP-портом 8080 виртуальной машины. Было ли это взаимодействие удачным?

2.2.6.Отобразите весь UDP-трафик, который присутствует в файле дампа.

2.2.7.Отобразите широковещательный трафик (с информацией уровня доступа к сети), который находится в файле дампа.

2.2.8.Отобразите ARP-трафик, который содержится в дампе (с информацией уровня доступа к сети).

3. В локальном сегменте сети (подсеть 192.168.6.0/24) сетевые станции не могут взаимодействовать друг с другом по протоколу IP. Изучить фрагмент вывода tcpdump и объяснить причину данной ситуации:

```
$ tcpdump -ne -r ex3.dump
reading from file ex3.dump, link-type EN10MB (Ethernet)
16:20:38.558302 00:00:00:00:00:01 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806),
  length 42: Request who-has 192.168.6.30 tell 192.168.6.5, length 28
16:20:38.580672 00:00:00:00:00:03 > 00:00:00:00:00:01, ethertype ARP (0x08
  length 42: Reply 192.168.6.30 is-at 00:00:00:00:00:03, length 28
16:20:38.639393 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
  length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 33887, seq 13
16:20:38.654419 00:00:00:00:00:02 > 00:00:00:00:00:01, ethertype ARP (0x08
  length 42: Reply 192.168.6.30 is-at 00:00:00:00:00:02, length 28
16:20:38.683649 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
  length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 5362, seq 618
16:20:38.709642 00:00:00:00:00:04 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806),
  length 42: Request who-has 192.168.6.10 tell 192.168.6.4, length 28
16:20:38.729719 00:00:00:00:00:05 > 00:00:00:00:00:04, ethertype ARP (0x08
  length 42: Reply 192.168.6.10 is-at 00:00:00:00:00:05, length 28
16:20:38.748087 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
  length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 4309, seq 567
16:20:38.770256 00:00:00:00:00:03 > 00:00:00:00:00:04, ethertype ARP (0x08
  length 42: Reply 192.168.6.10 is-at 00:00:00:00:00:03, length 28
16:20:38.803840 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
  length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 46657, seq 63
16:20:38.829548 00:00:00:00:00:04 > 00:00:00:00:00:05, ethertype IPv4 (0x08
  length 42: 192.168.6.4 > 192.168.6.10: ICMP echo request, id 6239, seq 433
16:20:38.861968 00:00:00:00:00:05 > 00:00:00:00:00:04, ethertype IPv4 (0x08
  length 42: 192.168.6.10 > 192.168.6.4: ICMP echo reply, id 0, seq 0, length 8
16:20:38.885832 00:00:00:00:00:06 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806),
  length 42: Request who-has 192.168.6.20 tell 192.168.6.21, length 28
16:20:38.914436 00:00:00:00:00:03 > 00:00:00:00:00:06, ethertype ARP (0x08
  length 42: Reply 192.168.6.20 is-at 00:00:00:00:00:03, length 28
16:20:38.953113 00:00:00:00:00:04 > 00:00:00:00:00:05, ethertype IPv4 (0x08
  length 42: 192.168.6.4 > 192.168.6.10: ICMP echo request, id 41141, seq 12
16:20:38.978185 00:00:00:00:00:10 > 00:00:00:00:00:06, ethertype ARP (0x08
  length 42: Reply 192.168.6.20 is-at 00:00:00:00:00:10, length 28
16:20:39.018692 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
  length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 15440, seq 26
16:20:39.037768 00:00:00:00:00:05 > 00:00:00:00:00:04, ethertype IPv4 (0x08
  length 42: 192.168.6.10 > 192.168.6.4: ICMP echo reply, id 0, seq 0, length 8
16:20:39.073481 00:00:00:00:00:01 > 00:00:00:00:00:03, ethertype IPv4 (0x08
```

```
    length 42: 192.168.6.5 > 192.168.6.30: ICMP echo request, id 1117, seq 654
16:20:39.197000 00:00:00:00:00:06 > 00:00:00:00:00:03, ethertype IPv4 (0x08
    length 70: 192.168.6.21.21519 > 192.168.6.20.53: 28042 A? www.tut.by. (2
16:20:39.252522 00:00:00:00:00:1a > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806),
    length 42: Request who-has 192.168.6.20 tell 192.168.6.100, length 28
16:20:39.270199 00:00:00:00:00:03 > 00:00:00:00:00:1a, ethertype ARP (0x08
    length 42: Reply 192.168.6.20 is-at 00:00:00:00:00:03, length 28
16:20:39.293142 00:00:00:00:00:10 > 00:00:00:00:00:1a, ethertype ARP (0x08
    length 42: Reply 192.168.6.20 is-at 00:00:00:00:00:10, length 28
16:20:39.408039 00:00:00:00:00:1a > 00:00:00:00:00:03, ethertype IPv4 (0x08
    length 72: 192.168.6.100.21455 > 192.168.6.20.53: 29961 A? www.aport.ru
16:20:39.546045 00:00:00:00:00:06 > 00:00:00:00:00:03, ethertype IPv4 (0x08
    length 70: 192.168.6.21.55962 > 192.168.6.20.53: 64123 A? www.tut.by. (2
16:20:39.685004 00:00:00:00:00:1a > 00:00:00:00:00:03, ethertype IPv4 (0x08
    length 72: 192.168.6.100.30522 > 192.168.6.20.53: 41887 A? www.aport.ru
```

Маршрутизация в Linux

Даны краткие сведения об использовании компьютера с Linux в качестве маршрутизатора и расширенных возможностях маршрутизации, в частности маршрутизация по IP-адресу источника; описаны особенности выбора локального исходящего IP-адреса. Приведены примеры маршрутизации виртуальных локальных сетей, объединения локальных сетей на основе маршрутизаторов, задания локального исходящего IP-адреса и добавление альтернативного маршрута для узла-источника на шлюзе

Содержание

Включение функции пересылки проходящих пакетов

Операционная система Linux позволяет не только организовать сложные схемы обработки сетевых пакетов локального узла, но и использовать компьютер с несколькими сетевыми интерфейсами (физическими и/или логическими) в качестве маршрутизатора (для передачи трафика удаленных узлов и сетей).

Примечание: В данном пособии рассматривается статическая маршрутизация — такой способ маршрутизации, при котором все необходимые маршруты устанавливаются администратором вручную (с помощью команд и конфигурационных файлов).

По умолчанию сетевая подсистема Linux обрабатывает пакеты, предназначенные данному узлу или созданные данным узлом; функция обработки пакетов, в которых не указан IP-адрес данной станции как источник или приемник (forwarding, пересылка пакетов) отключена. Для включения функции пересылки проходящих пакетов необходимо установить в 1 значение, записанное в файле ip_forward псевдофайловой системы proc:

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

Данная настройка будет действовать до перезагрузки компьютера. Для восстановления конфигурации нужно после перезагрузки выполнять указанную команду (указать ее в файле /etc/rc.local) или лучше установить в файле /etc/sysctl.conf следующий параметр:

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

Пример 10.1.Объединение виртуальных локальных сетей с помощью маршрутизатора на основе Linux.

Для того, чтобы в примере 5.6. "Постоянные сетевые конфигурации (на примере Debian/GNU Linux)" "Постоянные сетевые конфигурации" подсети 192.168.1.0/24 и 192.168.2.0/24 могли взаимодействовать друг с другом через указанный в примере компьютер Linux (выступает в качестве шлюза для данных подсетей) необходимо:

- на данном компьютере включить функцию пересылки пакетов одним из описанных выше способов;
- на всех компьютерах первой подсети указать в качестве шлюза для второй подсети адрес 192.168.1.4 (разумеется, также можно указать данный адрес как шлюз по умолчанию), а для компьютеров второй подсети - шлюз для первой подсети 192.168.2.4.

Таблица маршрутизации шлюза:

```
# ip route show
192.168.2.0/24 dev vlan2  proto kernel  scope link  src 192.168.2.4
192.168.1.0/24 dev vlan1  proto kernel  scope link  src 192.168.1.4
```

Эти записи добавляются в таблицу маршрутизации шлюза автоматически (например, если на интерфейсе vlan1 установлен IP-адрес 192.168.1.4/24, то система полагает, что сеть 192.168.1.0/24 непосредственно подключена к данному интерфейсу). Таким образом, если на интерфейс vlan1 шлюза приходят сетевые пакеты, в которых адрес назначения принадлежит второй подсети, то шлюз, согласно

таблицы маршрутизации, передает эти пакеты в сеть с интерфейса vlan2.

Пример 10.2.Объединение двух сетей на основе маршрутизаторов

Пусть необходимо обеспечить взаимодействие сетей 192.168.1.0/24 и 192.168.2.0/24 (рис. 10.1.), соединенных интерфейсами маршрутизаторов, установленных в каждой из сетей (интерфейсы маршрутизаторов в сетях имеют адреса 192.168.1.1 и 192.168.2.1 соответственно).

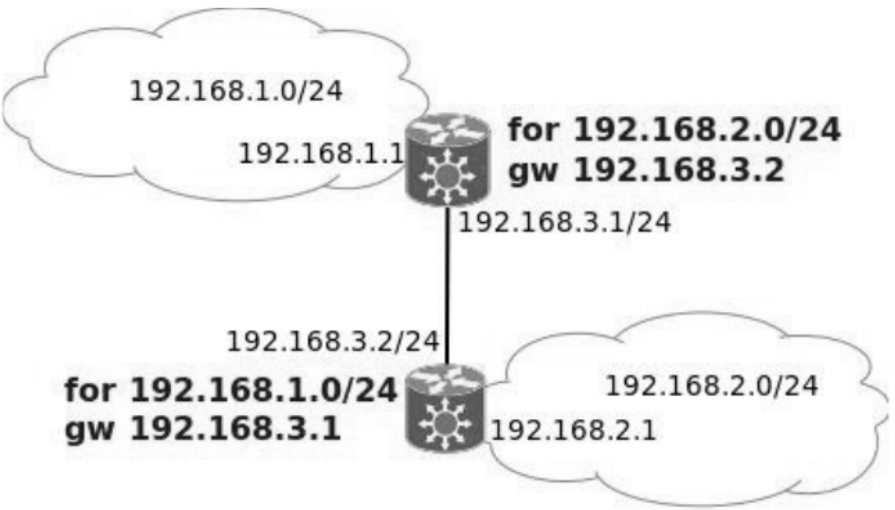

Рис. 10.1. Объединение двух сетей на основе маршрутизаторов

Для решения данной задачи (рис. 10.1.) необходимо установить на интерфейсы, связывающие маршрутизаторы друг с другом адреса из одной произвольно выбранной подсети (192.168.3.0/24, на первый — 192.168.3.1, на второй — 192.168.3.2) и добавить статические маршруты в таблицы маршрутизаторов (для первого маршрутизатора - для сети 192.168.2.0/24 через узел 192.168.3.2, для второго — для сети 192.168.1.0/24 через узел 192.168.3.1).

К примеру таблица маршрутизации первого маршрутизатора (при использовании Linux) будет иметь вид:

```
# ip route show
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.1
192.168.3.0/24 dev eth1  proto kernel  scope link  src 192.168.3.1
192.168.2.0/24 via 192.168.3.2 dev eth1
```

Последний маршрут здесь является статическим, так как должен быть задан администратором системы вручную.

Выбор IP-адреса для исходящих соединений

Выбор локального адреса для исходящих соединений в большинстве случаев системой осуществляется автоматически, исходя из имеющихся IP-адресов и таблицы маршрутизации (пример 10.3).

Примечание: Во многих случаях сетевое программное обеспечение позволяет указывать исходный адрес с помощью параметров.

Пример 10.3.Выбор исходящего IP-адреса

Пусть система имеет два интерфейса eth0(192.168.1.1/24) и eth1(192.168.56.102/24):

```
# ip addr show
...
2: eth0:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu 1500 qdisc pfif
    state UP qlen 1000 link/ether 08:00:27:23:22:97 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe23:2297/64 scope link
       valid_lft forever preferred_lft forever
3: eth1:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu 1500 qdisc pfif
    state UP qlen 1000 link/ether 08:00:27:fd:e5:aa brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global eth1
    inet6 fe80::a00:27ff:fefd:e5aa/64 scope link
       valid_lft forever preferred_lft forever
```

Маршрут по умолчанию у данной системы - 192.168.56.1:

```
# ip route show
```

```
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.1
192.168.56.0/24 dev eth1  proto kernel  scope link  src 192.168.56.102
default via 192.168.56.1 dev eth1
```

При такой конфигурации для исходящих соединений будет использоваться интерфейс eth1 и IP-адрес 192.168.56.102 (кроме соединений с узлами сети 192.168.1.0/24 — eth0 и IP-адрес 192.168.1.1). Ниже показан дамп сетевого пакета, отправленного командой `ping -c 1 192.168.3.4`:

```
# tcpdump -ne -i eth1 host 192.168.3.4
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
10:18:18.924084 08:00:27:fd:e5:aa > 0a:00:27:00:00:00, ethertype IPv4 (0x080
  length 98: 192.168.56.102 > 192.168.3.4: ICMP echo request, id 960, seq 1,
```

Однако, если добавить альтернативный маршрут для сети 192.168.3.0/24 через некоторый шлюз 192.168.1.254:

```
# ip route add 192.168.3.0/24 via 192.168.1.254
```

то, для пакетов, предназначенных узлу 192.168.3.4, будет использоваться интерфейс eth0 и исходящий адрес 192.168.1.1 (показан дамп сетевого пакета):

```
# tcpdump -ne -i eth0 host 192.168.3.4
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
10:23:33.393706 08:00:27:23:22:97 > 00:00:00:00:00:aa, ethertype IPv4 (0x08
  length 98: 192.168.1.1 > 192.168.3.4: ICMP echo request, id 968, seq 1, leng
```

Пример 10.4.Задание исходящего IP-адреса

Синтаксис команды `ip route` позволяет повлиять на выбор локального IP-адреса при соединении с удаленными системами. Для этого служит параметр `src` с указанием предпочитаемого IP-адреса (должен быть установлен на сетевом интерфейсе компьютера) для отправки пакетов на направление, определяемое в команде префиксом

маршрутизации.

Так, для указанной ниже конфигурации будет использоваться исходящий адрес 192.168.56.102 (кроме взаимодействия с узлами сети 192.168.1.0/24):

```
# ip addr show eth1
2: eth1:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu 1500 qdisc pfif
    link/ether 08:00:27:fd:e5:aa brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global eth1
    inet 192.168.1.10/24 brd 192.168.1.255 scope global eth1:1
    inet6 fe80::a00:27ff:fefd:e5aa/64 scope link
       valid_lft forever preferred_lft forever

# ip route show
192.168.1.0/24 dev eth1  proto kernel  scope link  src 192.168.1.10
192.168.56.0/24 dev eth1  proto kernel  scope link  src 192.168.56.102
default via 192.168.56.1 dev eth1
```

Для того, чтобы использовать исходный адрес 192.168.1.10 для соединения с узлами сети 192.168.3.0/24 следует использовать команду:

```
# ip route add 192.168.3.0/24 via 192.168.56.1 src 192.168.1.10 dev eth1:1
```

Таблица маршрутизации при этом будет иметь вид:

```
# ip route show
192.168.3.0/24 via 192.168.56.1 dev eth1  src 192.168.1.10
192.168.1.0/24 dev eth1  proto kernel  scope link  src 192.168.1.10
192.168.56.0/24 dev eth1  proto kernel  scope link  src 192.168.56.102
default via 192.168.56.1 dev eth1
```

Маршрутизация на основе правил и таблиц

Классические алгоритмы маршрутизации используют только адрес получателя в качестве аргумента при принятии решения какому шлюзу передавать пакет (маршрутизация по назначению, destination-routing). Однако в некоторых случаях необходимо использовать другие параметры сетевого пакета: адрес источника, вид и порт транспортного

протокола и т. д. (маршрутизация на основе политик, policy-routing) [45]. Маршрутизация на основе политик может использоваться при необходимости отправлять пакеты с разных IP-адресов, для отправки пакетов через разные интерфейсы для различных TCP-портов, балансировки нагрузки между различными Интернет-каналами и т. д. Маршрутизация по источнику (source-based routing) — вариант маршрутизации на основе политик, использующий адрес отправителя для принятия решения о маршрутизации.

Управление расширенными функциями маршрутизации в Linux осуществляется путем манипуляции множеством таблиц маршрутизации (ранее в изложении материала предполагалось, что таблица маршрутизации одна) и правилами использования этого множества (rules) [27]. Правила определяют порядок прохождения пакетов через таблицы. Если в данной таблице пакет соответствует направлению, указанному в какой-либо записи таблицы, то последующие возможные записи и таблицы игнорируются. Поэтому приоритет (номер) правила имеет важное значение.

Для управления списком правил используется команда `ip rule`. Принцип работы `ip rule` схож с другими командами `ip`. Применимы операции просмотра существующих правил (`list`), добавления нового правила (`add`), удаления правила (`del`). Далее в команде необходимо указать селектор выбора пакетов (на основе адреса\сети отправителя\получателя, TOS — типа обслуживания, внутренней маркировки пакетов и т. д.), а затем — действие, которое необходимо выполнить в случае, если пакет соответствует селектору, в частности, перенаправление в указанную таблицу маршрутизации (см. страницу руководства по команде `ip`).

По умолчанию в системе Linux присутствуют следующие таблицы маршрутизации:

```
# ip rule show
0:  from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

В данном случае любой пакет (`from all`) последовательно будет

проходить все таблицы (`local, main, default`) до первого совпадения какой-либо записи.

Таблица `local` содержит записи локального характера (маршрутизация "внутри" компьютера, широковещательные IP-адреса сегментов, в состав которых входит компьютер и т. д.). Данная таблица формируется автоматически и обычно не требует "ручного" редактирования. Пример таблицы `local`:

```
# ip route show table local

broadcast 192.168.1.0 dev eth1  proto kernel  scope link  src 192.168.1.10
broadcast 127.255.255.255 dev lo  proto kernel  scope link  src 127.0.0.1
local 192.168.56.102 dev eth1  proto kernel  scope host  src 192.168.56.102
broadcast 192.168.56.0 dev eth1  proto kernel  scope link  src 192.168.56.102
broadcast 192.168.1.255 dev eth1  proto kernel  scope link  src 192.168.1.10
local 192.168.1.10 dev eth1  proto kernel  scope host  src 192.168.1.10
broadcast 127.0.0.0 dev lo  proto kernel  scope link  src 127.0.0.1
broadcast 192.168.56.255 dev eth1  proto kernel  scope link  src 192.168.56.1
local 127.0.0.1 dev lo  proto kernel  scope host  src 127.0.0.1
local 127.0.0.0/8 dev lo  proto kernel  scope host  src 127.0.0.1
```

Таблица `main` включает основные записи о маршрутизации, с нею работают утилита `route` и команда `ip route` без указания таблицы (`ip route show` выводит содержимое таблицы `main`). Пример таблицы `main`:

```
# ip route show table main
192.168.1.0/24 dev eth1  proto kernel  scope link  src 192.168.1.10
192.168.56.0/24 dev eth1  proto kernel  scope link  src 192.168.56.102
default via 192.168.56.1 dev eth1
```

Таблица `default` обычно пустая.

Описание существующих в системе таблиц маршрутизации содержится в файле /etc/iproute2/rt_tables:

```
# cat /etc/iproute2/rt_tables
255  local
254  main
```

```
253 default
0 unspec
```

Для переопределения политики маршрутизации можно создавать дополнительные таблицы маршрутизации и правила отправки пакетов в данные таблицы (пример 10.5).

Пример 10.5.Добавление альтернативного маршрута для узла-источника на шлюзе.

Пусть шлюз локальной сети 192.168.1.0/24 имеет два сетевых интерфейса с IP-адресами 192.168.1.1/24 и 192.168.56.102/24 соответственно, маршрут по умолчанию — 192.168.56.1. Необходимо, чтобы шлюз отправлял трафик узла 192.168.1.10 по альтернативному маршруту 192.168.56.2.

Для решения данной задачи [27], во-первых, добавим в файл /etc/iproute2/rt_tables описание дополнительной таблицы маршрутизации (номер и имя таблицы выбираются произвольно, в данном случае `200` и `altroute`):

```
# echo 200 altroute >> /etc/iproute2/rt_tables
```

Во-вторых, добавим правило направления исходящих пакетов узла 192.168.1.10 в таблицу `altroute`:

```
# ip rule add from 192.168.1.10 lookup altroute
```

Список правил после выполнения данной команды:

```
# ip rule show
0: from all lookup local
32765: from 192.168.1.10 lookup altroute
32766: from all lookup main
32767: from all lookup default
```

В третьих, нужно в таблицу `altroute` добавить соответствующий маршрут:

```
# ip route add default via 192.168.56.2 dev eth1 table altroute
# ip route flush cache
```

Содержимое таблицы `altroute`:

```
# ip route show table altroute
default via 192.168.56.2 dev eth1
```

В результате на шлюзе исходящие пакеты узла 192.168.1.10 соответствующим правилом будут отправляться в таблицу `altroute`, в которой как маршрут по умолчанию указан узел 192.168.56.2. В тоже время исходящие пакеты других узлов сети 192.168.1.0/24 будут маршрутизироваться в соответствии с маршрутами, указанными в таблице `main`.

Ключевые термины

Статическая маршрутизация — такой способ маршрутизации, при котором все необходимые маршруты устанавливаются администратором вручную (с помощью команд и конфигурационных файлов).

Маршрутизация по назначению (destination-routing) — классический подход к маршрутизации, при котором используется адрес назначения в качестве аргумента при принятии решения какому шлюзу передавать пакет.

Маршрутизация на основе политик (policy-routing) — маршрутизация на основе различных параметров: адреса источника, вид и порт транспортного протокола и т. д.

Маршрутизация по источнику (source-based routing) — вариант маршрутизации на основе политик, использующий адрес отправителя для принятия решения о маршрутизации.

Краткие итоги

1. Сетевая подсистема Linux может обрабатывать как пакеты локального узла (по умолчанию), так и пакеты, не принадлежащие данному узлу (режим маршрутизатора). Обычно маршрутизатор

имеет два и более физических сетевых интерфейсов или хотя бы два и более IP-адресов из разных сетей. Необходимая для режима маршрутизатора функция пересылки пакетов (forwarding) требует дополнительной активации в настройках ядра Linux.

2. Выбор локального адреса для исходящих соединений в большинстве случаев системой осуществляется автоматически, исходя из имеющихся IP-адресов и таблицы маршрутизации. Синтаксис команды `ip route` позволяет повлиять на выбор локального IP-адреса при соединении с удаленными системами. Для этого служит параметр `src` с указанием предпочитаемого IP-адреса (должен быть установлен на сетевом интерфейсе компьютера) для отправки пакетов на направление, определяемое в команде префиксом маршрутизации. Кроме того, сетевое программное обеспечение обычно имеет настройки, позволяющие выбирать IP-адрес для исходящих подключений.
3. Управление расширенными функциями маршрутизации в Linux осуществляется путем манипуляций дополнительными таблицами маршрутизации и правилами, определяющими порядок прохождения пакетов через эти таблицы. Для управления списком правил используется команда `ip rule`.

Упражнения

В среде VirtualBox реализовать стенд для изучения маршрутизации, состоящий из двух виртуальных машин: основной ВМ и дополнительной ВМ.

Основная ВМ предназначена для реализации функции маршрутизации пакетов удаленных сетей. Непосредственно к ней подключены сети: 192.168.0.0/24, 192.168.1.0/24 и 192.168.2.0/24 ("физические" интерфейсы: eth1 с IP-адресом 192.168.0.1, eth2 с IP-адресом 192.168.1.1, eth3 с IP-адресом 192.168.2.1). Последние два интерфейса используются для связи с удаленными сетями (шлюз по умолчанию — 192.168.1.2, альтернативный маршрут — 192.168.2.2).

Примечание: Системы 192.168.1.2 и 192.168.2.2 не реализованы, имитация их присутствия осуществлена путем добавления в основную ВМ статических arp-записей для адресов 192.168.1.2 и 192.168.2.2 (при

наличии статической записи система не использует протокол arp и сразу отправляет фреймы на указанный MAC-адрес).

Дополнительная ВМ предназначена для тестирования настроек маршрутизатора и межсетевого экрана путем отсылки тестовых IP-пакетов. Сеть 192.168.0.0/24 объединяет виртуальные машины (для дополнительной ВМ — IP-адрес 192.168.0.2 на интерфейсе eth1, в качестве шлюза по умолчанию - 192.168.0.1).

Все сетевые адаптеры включены во внутреннюю сеть VirtualBox, которая работает в режиме концентратора.

Примечание: Так как при подключению к концентратору фреймы получают все устройства в сети, то при анализе трафика следует использовать команду `tcpdump` с указанием интерфейса мониторинга `(-i)` и не включая "беспорядочный" режим `(-p)`:

```
# tcpdump -nep -i eth2
```

Примечание: На практике включение несколько сетевых адаптеров системы в один сегмент физического или канального уровня используется для решения специальных задач: балансировки сетевой нагрузки, обеспечения отказоустойчивости и т.д. Для маршрутизации нескольких IP-подсетей, расположенных в одном широковещательном домене, достаточно одного сетевого адаптера и системной поддержки логических сетевых интерфейсов.

1.Установить основной IP-адрес (192.168.0.2 /24) и шлюз по умолчанию (192.168.0.1) на дополнительной ВМ:

```
# ip addr add dev eth1 local 192.168.0.2 /24 broadcast 192.168.0.255
# ip link set eth1 up
# ip route add default via 192.168.0.1
```

2.Установить в основной ВМ:

а)На интерфейс eth1 IP-адрес 192.168.0.1 /24

б)На интерфейс eth2 IP-адрес 192.168.1.1 /24

в)На интерфейс eth3 IP-адрес 192.168.2.1 /24

г)В качестве шлюза по умолчанию IP-адрес: 192.168.1.2

д)Внести статические записи в ARP-таблицу относительно адресов 192.168.1.2 и 192.168.2.2 (для их эмуляции):

```
# ip neighb  add 192.168.1.2 lladdr 00:00:00:00:00:AA dev eth2
# ip neighb  add 192.168.2.2 lladdr 00:00:00:00:00:BB dev eth3
```

е)Включить в ядре ОС функцию маршрутизации:

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

3.Проверить прохождения пакетов от узла 192.168.0.2 по маршруту по умолчанию:

а)В основной ВМ настроить прослушивание трафика на интерфейсе eth2, идущего на MAC-адрес узла 192.168.1.2:

```
# tcpdump -nep -i eth2
```

б)В дополнительной ВМ осуществить отправку тестовых ICMP-пакетов гипотетическому узлу 192.168.90.10 с IP-адреса 192.168.0.2:

```
# ping 192.168.90.10
```

в)Если все манипуляции проделаны правильно, то в дополнительной ВМ будет отображаться сообщение о недоступности узла 192.168.90.10, а в основной ВМ в TCPDump будут отображаться исходящие фреймы на узел 00:00:00:00:00:AA, содержащие пакеты с исходящим IP-адресом 192.168.90.10.

4.Установить статический маршрут до сети 192.168.100.0 /24 через альтернативный шлюз 192.168.2.2:

а) В основной ВМ:

```
# ip route add to 192.168.100.0/24 via 192.168.2.2
```

б)Запуская TCPDump на интерфейсах eth1 и eth2 основной ВМ и

используя команду ping в дополнительной ВМ убедиться, что трафик на сеть 192.168.100.0 /24 идет на интерфейс 192.168.2.2, а весь остальной трафик - на 192.168.1.2.

5.В основной ВМ установить маршрут до узла 192.168.101.11 через альтернативный шлюз 192.168.2.2.

6.В основной ВМ установить для пакетов узла 192.168.0.5 маршрут по умолчанию через альтернативный шлюз 192.168.2.2 (пример 10.5). В дополнительной ВМ установить дополнительный адрес 192.168.0.5 и проверить работоспособность установленного маршрута.

Межсетевое экранирование в Linux

Даны краткие сведения о подсистеме netfilter/iptables. Приведены примеры составления правил и взаимодействия правил и цепочек iptables в задачах фильтрации трафика

Содержание

Общие сведение о подсистеме netfilter

Подсистема netfilter представляет собой средство пакетной фильтрации в ядре Linux (начиная с версии 2.4) [46]. Утилита `iptables` используется для управления netfilter.

Примечание: Netfilter/iptables кроме создание межсетевых экранов на основе пакетной фильтрации (в том числе и с учетом состояния соединения), также позволяет реализовать различные сложные схемы манипуляции сетевыми пакетами, в частности трансляцию сетевых адресов (NAT) для разделяемого доступа в Интернет и организации прозрачных прокси (см. "Обеспечение доступа в сеть Интернет").

Сетевые пакеты в подсистеме netfilter проходят последовательность цепочек (chain). Каждая цепочка включает заданный набор таблиц (table), содержащих упорядоченные списки правил. Каждое правило содержит параметры выбора и выполняемое действие при соответствии содержимого пакета параметрам.

Существует пять типов стандартных цепочек, встроенных в систему [47]:

- PREROUTING — для изначальной обработки входящих пакетов
- INPUT — для входящих пакетов, адресованных непосредственно локальному компьютеру
- FORWARD — для проходящих (маршрутизируемых) пакетов

- OUTPUT — для пакетов, создаваемых локальным компьютером
- POSTROUTING — для окончательной обработки исходящих пакетов

Цепочки организованы в четыре таблицы рис. 11.1 [47]:

- *raw* — просматривается до передачи пакета системе определения состояний. Содержится в цепочках PREROUTING и OUTPUT.
- *mangle* — содержит правила модификации заголовка IP? пакетов. Среди прочего, поддерживает изменения полей TTL и TOS, и маркеров пакета. Содержится во всех стандартных цепочках.
- *nat* — используется для трансляции сетевых адресов. Содержится в цепочках PREROUTING, OUTPUT, и POSTROUTING.
- `filter` — основная таблица, предназначенная для фильтрации сетевых пакетов; используется по умолчанию если название таблицы не указано. Содержится в цепочках INPUT, FORWARD, и OUTPUT.

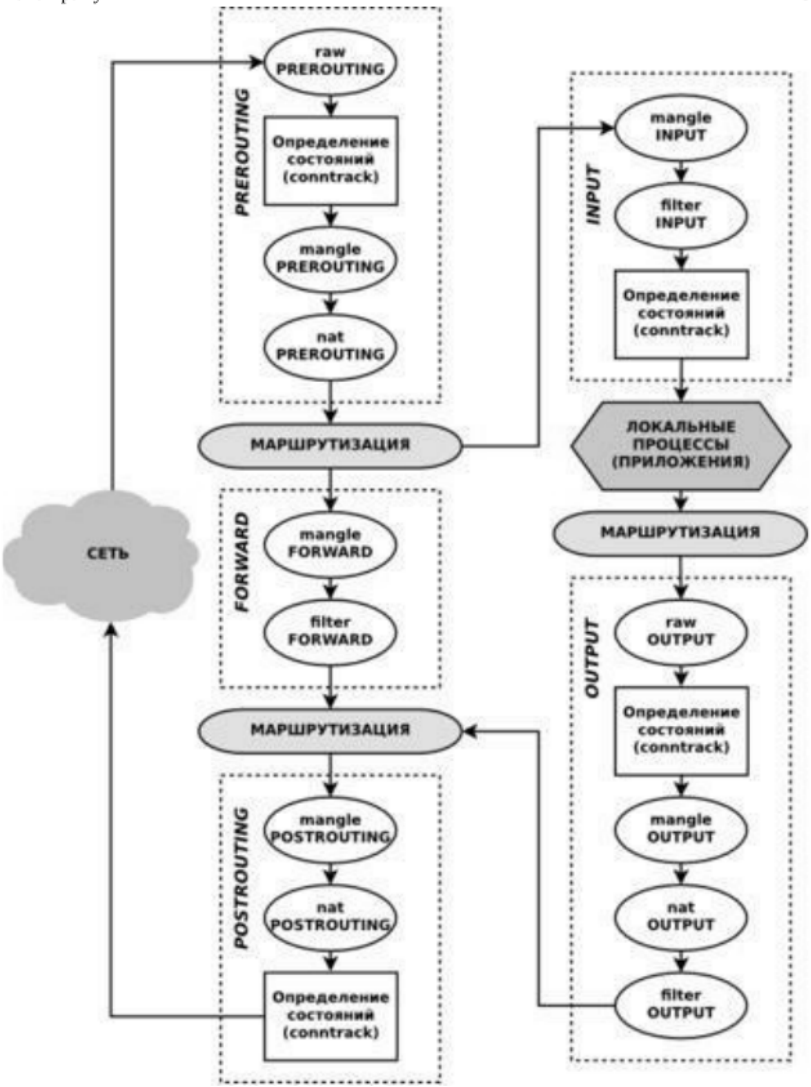

Рис. 11.1. Прохождение пакета в подсистеме netfilter (взято из wikipedia.org)

Следует отметить, что одноименные таблицы каждой цепочки независимы.

Входящий пакет (рис 11.1) начинает обрабатываться с цепочки PREROUTING в таблицах `raw`, `conntrack` (определение состояний) и `mangle`. Затем он обрабатывается правилами таблицы `nat` данной цепочки. На этом этапе проверяется, необходима ли

модификация адреса получателя пакета (DNAT). Важно сменить адрес получателя на данном этапе, так как маршрут пакета определяется сразу после того, как он покинет цепочку PREROUTING.

Далее возможны два варианта:

1. Если целью пакета является этот компьютер, то пакет будет отправлен в цепочку INPUT ; после маршрутизации, он обрабатывается правилами цепочек INPUT. В случае прохождения цепочек пакет передается приложению.
2. Если целью пакета является другой компьютер, то пакет фильтруется правилами цепочки FORWARD таблиц *mangle и* `filter`, а затем к нему применяются правила цепочки POSTROUTING. На данном этапе можно использовать SNAT/MASQUERADE (подмена источника/маскировка). После этих действий пакет будет отправлен в сеть.

Третий вариант - когда приложение на компьютере, отвечает на запрос или отправляет собственный пакет, то он обрабатывается цепочкой OUTPUT таблиц *raw*, `conntrack` *и* `filter`. Затем к нему применяются правила цепочки OUTPUT таблицы *nat*, для определения, требуется ли использовать DNAT (модификация адреса получателя), пакет фильтруется цепочкой OUTPUT таблицы `filter` и выпускается в цепочку POSTROUTING. В случае успешного прохождения POSTROUTING пакет выходит в сеть.

Непосредственно для фильтрации используются таблицы `filter`. Поэтому в рамках данной темы важно понимать, что для фильтрации пакетов, предназначенных данному узлу необходимо модифицировать таблицу `filter` цепочки INPUT, для проходящих пакетов — цепочки FORWARD, для пакетов, созданных данным узлом — OUTPUT.

Управление правилами iptables

Команда `iptables` позволяет редактировать правила таблиц `netfilter`. Каждое правило представляет собой запись, содержащую в себе параметры отбора (критерии), определяющие, подпадает ли пакет под данное правило, и действие, которое необходимо выполнить в

случае соответствия параметрам отбора [48].

В общем виде правила записываются в виде:

```
iptables [-t table] command [match] [target/jump]
```

По умолчанию используется таблица `filter`, если же необходимо указать другую таблицу, то следует использовать спецификатор `-t` с указанием имени таблицы. После имени таблицы указывается команда, определяющая действие: вставить правило, или добавить правило в конец цепочки, или удалить правило и т. п. `Match` задает параметры отбора. `Target` указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле: передать пакет в другую цепочку правил, "сбросить" пакет, выдать на источник сообщение об ошибке и т. п.

Примечание: для работы с `iptables` требуются права суперпользователя.

Для просмотра существующих правил, команда `iptables` используется с ключом `-L`. Так, для просмотра всех цепочек таблицы `filter` (`-v` — для более детального отображения):

```
# iptables -L -v
```

Примечание: Далее по тексту данной темы, если не названа таблица, то имеется в виду таблица `filter`.

Для просмотра цепочки FORWARD нужно указать имя цепочки:

```
# iptables -v -L FORWARD
```

Сбросить все правила `(-F)` и удалить определенные пользователем цепочки `(-X)`:

```
# iptables -F
# iptables -X
```

Для добавления правила в цепочку используется ключ `-A`. Например, чтобы добавить правило в цепочку POSTROUTING таблицы *`nat`*:

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Для того, чтобы добавить правило в цепочку FORWARD:

```
# iptables -A FORWARD -i eth0 --destination 192.168.1.0/24  -j ACCEPT
```

Ниже в виде таблицы приведены основные параметры отбора пакетов [48]:

Таблица 1.1.

Параметр	Описание	Пример
`--protocol` (сокращено `-p`)	Определяет протокол. Опции `tcp, udp, icmp`, или любой другой протокол определенный в /etc/protocols	`iptables -A INPUT -p tcp`
`--source (-s)`	IP адрес источника пакета. Может быть определен несколькими путями. • Одиночный хост: `host.domain.tld`, или IP адрес: 10.10.10.3 • Пул-адресов (подсеть): 10.10.10.3/24 или 10.10.10.3/255.255.255.0	`iptables -A INPUT -s 10.10.10.3`
`--destination (-d)`	IP адрес назначения пакета. Может быть определен несколькими путями - смотри — `source`	`iptables -A INPUT --destination 192.168.1.0/24`
`--source-port (--sport)`	Порт источник, возможно только для протоколов — `protocol tcp`, или — `protocol udp`	`iptables -A INPUT --protocol tcp --source-port 25`
`--destination-port (--`	Порт назначения, возможно только для протоколов — `protocol tcp`, или —	`iptables -A INPUT --protocol udp --destination-`

dport)	protocol udp	port 67
--state	Состояние соединения. Доступно, если модуль 'state' загружен с помощью '-m state'. Доступные опции: NEW (Все пакеты устанавливающие новое соединение) ESTABLISHED (Все пакеты, принадлежащие установленному соединению) RELATED (Пакеты, не принадлежащие установленному соединению, но связанные с ним. Например - FTP в активном режиме использует разные соединения для передачи данных. Эти соединения связаны.) INVALID (Пакеты, которые не могут быть по тем или иным причинам идентифицированы. Например, ICMP ошибки не принадлежащие существующим соединениям)	iptables -A INPUT -m state --state NEW,ESTABLISHED
--in-interface	Определяет интерфейс, на который прибыл пакет. Полезно для NAT и машин с	iptables -t *nat* -A PREROUTING -

(сокращенно `-i`)	несколькими сетевыми интерфейсами	`-in-interface eth0`
`--out-interface` (сокращенно `-o`)	Определяет интерфейс, с которого уйдет пакет. Полезно для NAT и машин с несколькими сетевыми интерфейсами	`iptables -t nat -A POSTROUTING --out-interface eth1`
`--tcp-flags`	Определяет TCP флаги пакета. Содержит 2 параметра: Список флагов которые следует проверить и список флагов которые должны быть установлены	
`--syn`	Сокращение для `'--tcp-flags SYN,RST,ACK SYN'` . Поскольку проверяет TCP флаги, используется с — `protocol tcp`. В примере показан фильтр для пакетов с флагом `NEW`, но без флага `SYN`. Обычно такие пакеты должны быть выброшены (`DROP`).	`iptables -A INPUT --protocol tcp ! --syn -m state --state NEW`

Действие, которое система выполнит, если пакет в одной из цепочек удовлетворяет условию, устанавливается с помощью ключа `-j (--jump)` . Можно также передать пакет в другую цепочку.

Стандартные действия:

1. `-ACCEPT` - пакет покидает данную цепочку и передается в следующую.
2. `-DROP` - отбросить удовлетворяющий условию пакет.
3. `-REJECT` - отбросить пакет, отправив отправителю ICMP-сообщение.
4. `-LOG` - протоколировать пакет.

5. `-RETURN` - возвратить пакет в предыдущую цепочку.
6. `-SNAT` - применить трансляцию адреса источника в пакете. Может использоваться только в цепочках POSTROUTING и OUTPUT таблицы *`nat`*.
7. `-DNAT` - применить трансляцию адреса назначения в пакете. Может использоваться только в цепочке POSTROUTING таблицы *`nat`*.
8. *`-MASQUERADE`* — используется вместо SNAT при наличии соединения с динамическим IP.
9. `-MARK` — используется для установки меток на пакеты.

```
# iptables -A input -p tcp -s 192.168.0.0/16 --dport 80 -j ACCEPT
```

Листинг 11.1. Правило, разрешающее подключения к локальному HTTP-серверу из заданной подсети

```
# iptables -A FORWARD -s 192.168.1.0/24 -d 192.168.9.9 -j DROP
```

Листинг 11.2. Правило, запрещающее проходящий трафика от сети 192.168.1.0/24 к узлу 192.168.9.9

Взаимодействие правил и цепочек

Для каждой таблицы в цепочке определена политика обработки пакетов по умолчанию (тех, которые не соответствуют ни одному правилу данной таблицы). Политика по умолчанию может быть, например, разрешающая (`ACCEPT`), т. е. передавать пакеты в следующую таблицу, или запрещающая (`DROP`), т. е. уничтожать пакеты (пример 11.3).

Для того, чтобы установить запрещающую политику (уничтожения без сообщения адресату) для проходящих пакетов, не удовлетворяющих ни одному правилу нужно использовать `iptables` с ключом `-P` с указанием цепочки и действия:

```
# iptables -P FORWARD DROP
# iptables -L FORWARD
Chain FORWARD (policy DROP)
target     prot opt source               destination
```

Листинг 11.3. Установка политики по умолчанию для цепочки

Примечание: Указанная в пример 11.3 команда фактически устанавливает запрещающую политику по умолчанию только в таблице filter цепочки FORWARD. В таблице mangle данной цепочки политика по умолчанию не изменилась:

```
# iptables -t mangle -L FORWARD
Chain FORWARD (policy ACCEPT)
target     prot opt source
```

Так как политику фильтрации пакетов в цепочке принято определять в таблице `filter`, то в контексте фильтрации имеется ввиду, что задается политика по умолчанию для всей цепочки (здесь и далее по тексту).

При добавлении правил `iptables` следует учитывать то, что их порядок в цепочке имеет значение (пример 11.4). Чтобы вставить правило в произвольное место существующей цепочки, укажите ключ `-I`, затем название этой цепочки и позицию (1,2,3,...,n), в которой должно располагаться правило.

Пусть существует разрешительная политика по умолчанию для входящего локального трафика и запретительное правило на доступ к локальному HTTP-серверу из сети 192.168.0.0/16:

```
# iptables -L INPUT -v
Chain INPUT (policy ACCEPT 7 packets, 508 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       tcp  --  any    any     192.168.0.0/16       anywhere             t
```

Листинг 11.4. Добавление правила в заданную позицию в цепочке

Для того, чтобы сделать исключение для узла 192.168.1.5 (входящему в сеть 192.168.0.0/16), соответствующее разрешительное правило, должно быть достижимо, поэтому его необходимо разместить перед запретительным:

```
# iptables -I INPUT 1 -p tcp -s 192.168.1.5 --dport 80 -j ACCEPT
# iptables -L INPUT -v
Chain INPUT (policy ACCEPT 7 packets, 508 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

```
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  any    any     192.168.1.5          anywhere             t
    0     0 DROP       tcp  --  any    any     192.168.0.0/16       anywhere             t
```

Возможность сетевого взаимодействия с участием данного узла или узлов, трафик которых проходит через данный узел, определяется действием на сетевые пакеты всех цепочек, через которые они проходят (Пример 11.5).

Для того, чтобы обеспечить доступ узла 192.168.1.5 к локальному HTTP-серверу при запретительной политики по умолчанию цепочек INPUT и OUTPUT, соответствующие разрешительные правила необходимо добавить как в цепочку INPUT, так и в цепочку OUTPUT:

```
# iptables -A INPUT -p tcp -s 192.168.1.5 --dport 80 -j ACCEPT
# iptables -A OUTPUT -p tcp -d 192.168.1.5 -j ACCEPT
# iptables -L INPUT -v
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
 4      336   ACCEPT     all  --  lo     any     anywhere             anywhere
 1536  123K ACCEPT     all  --  any    any     192.168.56.1         anywhere
    0     0       ACCEPT     tcp  --  any    any     192.168.1.5          anywhere
# iptables -L OUTPUT -v
Chain OUTPUT (policy DROP 8 packets, 564 bytes)
 pkts bytes target     prot opt in     out     source               destination
    4      336 ACCEPT     all  --  any    lo      anywhere             anywhere
 1050  106K ACCEPT     all  --  any    any     anywhere             192.168.56.1
    0        0 ACCEPT     all  --  any    any     anywhere             192.168.1.5
```

Примечание: При запретительной политике по умолчанию для цепочек INPUT и OUTPUT необходимо разрешить трафик интерфейса *loopback* (lo):

```
# iptables -A OUTPUT -o lo -j ACCEPT
# iptables -A INPUT -i lo -j ACCEPT
```

Краткие итоги

1. Подсистема netfilter, входящая в состав ядра Linux, используется для

схем манипуляции сетевыми пакетами, в т.ч. трансляции сетевых адресов (NAT). Для управления `netfilter` используется утилита `iptables`.

2. Сетевые пакеты в подсистеме `netfilter` проходят последовательность цепочек (*`chain`*). Каждая цепочка включает заданный набор таблиц, содержащих упорядоченные списки правил. Каждое правило содержит параметры выбора и выполняемое действие при соответствии содержимого пакета параметрам.
3. Для фильтрации пакетов, предназначенных данному узлу используются правила таблицы `filter` цепочки INPUT, для проходящих пакетов — цепочки FORWARD, для пакетов, созданных данным узлом — OUTPUT.
4. Для каждой цепочки задается политика для пакетов, не удовлетворяющих ни одному правилу цепочки. При добавлении правил `iptables` следует учитывать то, что их порядок в цепочке имеет значение. Возможность сетевого взаимодействия с участием данного узла или узлов, трафик которых проходит через данный узел, определяется действием на сетевые пакеты всех цепочек, через которые они проходят.

Упражнения

Используя стенд на основе VirtaulBox, описанный в упражнении "Маршрутизация в Linux", настроить фильтрацию трафика на маршрутизаторе:

1. Разрешить IP-трафик от сети 192.168.0.0/24 к маршрутизатору на интерфейсе eth1 ; весь остальной входящий трафик маршрутизатора на интерфейсе eth1 запретить.
2. Разрешить проходящий IP-трафик для сети 192.168.0.0/24; проходящий трафик других сетей запретить.
3. Запретить ICMP-трафик от сети 192.168.0.0/24 к сети 192.168.9.0/24.
4. Запретить проходящий IP-трафик от сети 192.168.0.0/24 к сети 192.168.10.0/24.
5. Запретить IP-трафик от сети 192.168.0.0/24 к узлу 192.168.11.11

1.Реализовать п. 1 (основная ВМ):

```
# iptables -P INPUT DROP
# iptables -A INPUT -i lo -p all -j ACCEPT
# iptables -A INPUT -i eth1 --source 192.168.0.0/24 -j ACCEPT
```

2.Реализовать п. 2:

```
# iptables -P FORWARD DROP
# iptables -A FORWARD -i eth1 --source 192.168.0.0/24 -j ACCEPT
# iptables -A FORWARD -i eth2 --destination 192.168.0.0/24 -j ACCEPT
# iptables -A FORWARD -i eth3 --destination 192.168.0.0/24 -j ACCEPT
```

3.Реализовать п. 3.

```
# iptables -I FORWARD 1 -i eth1 --source 192.168.0.0/24 --destination
  192.168.9.0/24 --protocol icmp -j DROP
```

4.Реализовать пп. 4 и 5. самостоятельно.

Обеспечение доступа в сеть Интернет

Даны краткие сведения о распространенных технологиях подключения к сети Интернет. Приведены примеры настройки PPPoE-сервера и клиента; настройки 3G-модема; использования технологий NAT для организации доступа к сети Интернет и публикации ресурса локальной сети в Интернет, также настройки PPTP-клиента и DNS-прокси

Содержание

Подключение к сети Интернет

В настоящее время для пользователей и организаций существует множество возможностей подключения к инфраструктуре провайдеров Интернет-услуг. Для доступа к глобальной сети во многих случаях со стороны клиента необходимо устанавливать специальное оборудование (модемы и т.д.), настраивать логические подключения с использованием технологий виртуальных частных сетей (VPN — Virtual Private Network) и т.д.

Примечание: Большинство технологий VPN используют инкапсуляцию протокола точка-точка PPP (Point-to-Point Protocol) – логического механизма, обеспечивающего функционирование сетевых протоколов (IP, IPX, NetBEUI) на последовательных каналах передачи (соединениях между двумя узлами).

Широко используемыми способами подключения к сети Интернет являются:

- Технологии группы xDSL на основе существующей инфраструктуры проводной телефонной сети. В частности широкое распространение получила технология ADSL,

позволяющая использовать один телефонный провод для стационарной телефонной связи и доступа в Интернет. В качестве оконечного оборудования в ADSL используется модем, подключаемый к ADSL-коммутатору провайдера, а также для разделения голосового потока и потока данных обычно необходим сплиттер (пассивное оборудование). ADSL-модем обычно имеет один или несколько Ethernet-интерфейсов для включения в локальную сеть, подключения к маршрутизатору локальной сети или серверу доступа.

Для организации доступа к Интернет по технологии xDSL провайдеры обычно используют протокол PPPoE (Point-to-point protocol over Ethernet) — протокол передачи кадров PPP по сетям, построенным по технологии Ethernet (т.е. работающий на уровне доступа к сети). Данный подход позволяет использовать традиционное ПО последовательных линий передачи и организовать типичное соединение с логином и паролем для доступа в Интернет, т.е. осуществляется виртуальный "звонок" на Ethernet-машину и устанавливается соединение точка-точка (пример 12.1).

- Мобильный доступ в Интернет на основе технологий 3G. Услуга предоставляется операторами мобильной связи. В качестве оконечного оборудования используется GSM-модем, встроенный в мобильный телефон или сетевое оборудование, а также в виде отдельного устройства с USB или PCMCIA-интерфейсами (пример 12.2).
- Беспроводный доступ Wi-Fi/WiMAX. (на основе группы стандартов IEEE 802.11 и 802.16). В качестве оконечного оборудования выступает соответствующий беспроводный адаптер.
- Доступ в Интернет по технологии Ethernet. Некоторые провайдеры строят свою инфраструктуру таким образом, что подключение конечного абонента осуществляется по технологии Ethernet с использованием витой пары или оптоволоконного кабеля. В качестве оконечного оборудования выступает соответствующий Ethernet-адаптер.

В некоторых случаях технологии доступа могут также использоваться

совместно с протоколами туннелирования сетевого уровня, такими как GRE (Generic Routing Encapsulation) или PPTP (Point-to-point tunneling protocol). Пример использования PPTP VPN приведен в п. 12.4.

Протокол PPPoE работает в Ethernet-среде (широковещательном домене) на основе MAC-адресов. Клиент посылает широковещательный Ethernet фрейм, на который должен ответить PPPoE-сервер (адрес отправителя — свой MAC-адрес, адрес получателя — FF:FF:FF:FF:FF:FF и тип фрейма — PPPoE Discovery). PPPoE сервер посылает клиенту ответ (адрес отправителя — свой MAC-адрес, адрес получателя — MAC-адрес клиента и тип фрейма — PPPoE Discovery). Если в сети несколько PPPoE серверов, то все они посылают ответ. Клиент выбирает подходящий сервер и посылает ему запрос на соединение. Сервер посылает клиенту подтверждение с уникальным идентификатором сессии, все последующие фреймы в сессии будут иметь этот идентификатор. Таким образом создается виртуальный канал для PPP-подключения.

Для настройки простейшего PPPoE-сервера в Debian необходимо:

1. Установить пакеты *ppp* и *pppoe*.
2. Создать файл /etc/ppp/pppoe-server-options (файл настроек PPPoE-сервера) и вписать в него следующие данные:

```
logfile /var/log/pppoe.log
debug
mtu 1472
mru 1472
auth
login
default-asyncmap
ktune
lcp-echo-interval 20
lcp-echo-failure 2
proxyarp
```

Листинг 12.1. Настройка PPPoE-сервера и клиента в Debian

3. Для проверки правильности настройки сервера PPPoE создать тестовую учетную запись. Для этого открыть файл /etc/ppp/chap-

secrets и записать в нее одну строку:

```
test * pass *
```

4. Запустить PPPoE-сервер. Пусть на сервере два сетевых интерфейса: eth1 (с установленным IP-адресом и шлюзом по умолчанию — для доступа к Интернет), а интерфейс, к которому будут подключаться PPPoE-клиенты — eth0 (без IP-адреса).

```
# pppoe-server -I eth0 -R 192.168.0.33 -L 192.168.0.1 -O /etc/ppp/ppp
```

Параметр `-I` позволяет указать на специфический интерфейс (в данном случае - eth0), при помощи `-L` указываем локальный адрес сервера внутри туннеля. Параметр `-R` указывает начальный адрес выдаваемый клиенту.

5. Для тестирования PPPoE-подключения необходима еще одна система с установленным PPPoE-клиентом, подключенная в тот же сегмент, что и интерфейс eth0 сервера (профиль dsl-provider и соответствующий пароль в Debian указаны по умолчанию):

```
# pon dsl-provider
```

Для тестирования работоспособности PPPoE-соединения:

```
# ping 192.168.0.1
```

Для отключения туннеля:

```
# poff
```

Предварительно необходимо установить пакеты usb-modeswitch (для отключения режима USB-накопителя) и ppp (для организации WAN-подключения):

```
# apt-get install usb-modeswitch
# apt-get install ppp
```

Листинг 12.2. Настройка 3G-модема в Debian (HUAWEI E173)

Далее при включении данного USB-гаджета система автоматически

обнаружит и загрузит необходимые модули ядра (приведен фрагмент системного протокола):

```
11:38 3gtest kernel: [ 1267.612036] usb 1-2: new high speed USB device using
  ehci_hcd and address 3
11:38 3gtest kernel: [ 1267.746681] usb 1-2: New USB device found, idVendor
  idProduct=1446
11:38 3gtest kernel: [ 1267.746686] usb 1-2: New USB device strings: Mfr=3, I
  SerialNumber=0
11:38 3gtest kernel: [ 1267.746690] usb 1-2: Product: HUAWEI Mobile
11:38 3gtest kernel: [ 1267.746693] usb 1-2: Manufacturer: HUAWEI Technolo
11:38 3gtest kernel: [ 1267.746822] usb 1-2: configuration #1 chosen from 1
  choice
11:38 3gtest kernel: [ 1267.748801] scsi6 : SCSI emulation for USB Mass Stora
11:38 3gtest kernel: [ 1267.749150] scsi7 : SCSI emulation for USB Mass Stora
11:39 3gtest usb_modeswitch: switching 12d1:1446 (HUAWEI Technology: HU
11:39 3gtest kernel: [ 1268.451065] usb 1-2: USB disconnect, address 3
11:43 3gtest kernel: [ 1272.620027] usb 1-2:
  new high speed USB device using ehci_hcd and address 4
11:43 3gtest kernel: [ 1272.754434] usb 1-2: New USB device found, idVendor
  idProduct=140c
11:43 3gtest kernel: [ 1272.754439] usb 1-2: New USB device strings: Mfr=3,
  Product=2, SerialNumber=0
11:43 3gtest kernel: [ 1272.754443] usb 1-2: Product: HUAWEI Mobile
11:43 3gtest kernel: [ 1272.754445] usb 1-2: Manufacturer: HUAWEI Technolo
11:43 3gtest kernel: [ 1272.754569] usb 1-2: configuration #1 chosen from 1 ch
11:43 3gtest kernel: [ 1272.758091] scsi12 : SCSI emulation for USB Mass Sto
11:43 3gtest kernel: [ 1272.758581] scsi13 : SCSI emulation for USB Mass Sto
11:43 3gtest kernel: [ 1272.800270] usbcore: registered new interface driver usb
11:43 3gtest kernel: [ 1272.800637] USB Serial support registered for generic
11:43 3gtest kernel: [ 1272.800820] usbcore: registered new interface driver usb
11:43 3gtest kernel: [ 1272.800825] usbserial: USB Serial Driver core
11:43 3gtest kernel: [ 1272.825756] USB Serial support registered for GSM mo
11:43 3gtest kernel: [ 1272.826537] option 1-2:1.0:
  GSM modem (1-port) converter detected
11:43 3gtest kernel: [ 1272.826722] usb 1-2:
  GSM modem (1-port) converter now attached to ttyUSB0
11:43 3gtest kernel: [ 1272.826752] option 1-2:1.1:
  GSM modem (1-port) converter detected
```

```
11:43 3gtest kernel: [ 1272.827252] usb 1-2:
  GSM modem (1-port) converter now attached to ttyUSB1
11:43 3gtest kernel: [ 1272.827401] option 1-2:1.2:
  GSM modem (1-port) converter detected
11:43 3gtest kernel: [ 1272.828720] usb 1-2:
  GSM modem (1-port) converter now attached to ttyUSB2
11:43 3gtest kernel: [ 1272.828757] option 1-2:1.3:
  GSM modem (1-port) converter detected
11:43 3gtest kernel: [ 1272.831879] usb 1-2:
  GSM modem (1-port) converter now attached to ttyUSB3
11:43 3gtest kernel: [ 1272.831946] usbcore: registered new interface driver opt
11:43 3gtest kernel: [ 1272.831952] option: v0.7.2:USB Driver for GSM moder
11:44 3gtest usb_modeswitch: switched to 12d1:140c (HUAWEI Technology: H
11:48 3gtest kernel: [ 1277.758093] scsi 12:0:0:0:
  CD-ROM            HUAWEI   Mass Storage     2.31 PQ: 0 ANSI: 2
11:48 3gtest kernel: [ 1277.758218] scsi 13:0:0:0:
  Direct-Access     HUAWEI   SD Storage       2.31 PQ: 0 ANSI: 2
11:48 3gtest kernel: [ 1277.758971] sd 13:0:0:0: Attached scsi generic sg2 type
11:48 3gtest kernel: [ 1277.763536] sd 13:0:0:0: [sdb] Attached SCSI removab
11:48 3gtest kernel: [ 1277.766333] sr1: scsi-1 drive
11:48 3gtest kernel: [ 1277.766687] sr 12:0:0:0: Attached scsi generic sg3 type 5
```

В данном случае создается четыре интерфейса модема (ttyUSB0-3), интерфейс виртуального CDROM (записано программное обеспечение для Windows) и интерфейс FLASH-памяти.

Далее необходимо в файле /etc/ppp/peers/3g настройки PPP-соединения, а в файле /etc/ppp/3g — для модема команды инициализации и набора номера:

```
# cat /etc/ppp/peers/3g
connect "usr/sbin/chat -v -f /etc/ppp/3g1" /dev/ttyUSB0
460800
#115400
crtscts
#debug
#logfile /var/log/1.log
noauth
noipdefault
```

```
usepeerdns
defaultroute
user  "web1"
password  "web"

# cat /etc/ppp/3g
TIMEOUT 35
ECHO ON
ABORT '\nBUSY\r'
ABORT '\nERROR\r'
ABORT '\nNO ANSWER\r'
ABORT '\nNO CARRIER\r'
ABORT '\nNO DIALTONE\r'
ABORT '\nRINGING\r\n\r\nRINGING\r'
ABORT '\nUsername/pASSWORD Incorrect\r'
''  \rAT
OK   'AT+CGDCONT=1,\ "IP\",\"web1.velcom.by\"' '
OK  'ATD*99#'
CONNECT ''
```

Подключение:

```
# pppd call 3g
```

Примечание: Если SIM-карта использует PIN-код:

```
# cat /var/log/ppp-connect-errors
AT
OK
AT+CGDCONT=1, "IP" , "web1.velcom.by"
+CME ERROR: SIM PIN required
```

то его необходимо один раз передать модему при инициализации:

```
OK  'AT+CPIN=8971'
```

Особенности доступа к сети Интернет на сетевом уровне

Обычно локальные сети строятся с использованием локальных (автономных) диапазонов IP-адресов (10.0.0.0 /8, 172.16.0.0-172.31.0.0, 192.168.0.0 /16). Для взаимодействия с сетью Интернет используется один IP-адрес или небольшой диапазон IP-адресов, предоставляемый провайдером Интернет-услуг.

Если нет необходимости публиковать локальные информационные ресурсы в Интернет, то для исходящих соединений узлов локальной сети с узлами Интернет может использоваться Интернет-адрес провайдера, задаваемый динамически на внешнем интерфейсе (WAN-интерфейсе) модема, маршрутизатора или прокси-сервера. Для публикации локальных информационных ресурсов в сети Интернет необходим хотя бы один статический реальный IP-адрес. Для взаимодействия узлов локальной сети с сетью Интернет на сетевом и транспортном уровнях может использоваться трансляции сетевых адресов (NAT, Network Address Translation), а на прикладном уровне — технологии прокси.

Примечание: Прокси-сервер прикладного уровня осуществляет функцию посредника между компьютерами локальной сети и серверами Интернет. Обычно прокси-сервер реализует ограниченный набор прикладных протоколов (HTTP, HTTPS, SOCKS и т. д.). Использование прокси-сервера для WEB-трафика позволяет реализовать кэширование данных, получаемых из Интернет, фильтровать содержимое и управлять доступом к ресурсам Интернет. При этом, Интернет-обозреватель или иная программа, осуществляющая работу с данным прокси-сервером должна быть настроена (указан адрес прокси и порт на котором он принимает запросы). Примером WEB-прокси является кэширующий прокси-сервер Squid [49], поддерживающий протоколы HTTP, HTTPS, FTP и другие (см. п 8.5. "Настройка некоторых сетевых служб").

Доступ в Интернет узлов локальной сети (рис. 12.1) может осуществляться с помощью трансляции адреса источника (source NAT). На граничном устройстве (маршрутизаторе, модеме или межсетевом экране, минимально имеющем адрес локальной сети и Интернет-адрес), при взаимодействии с узлом Интернет адрес локального узла подменяется Интернет-адресом граничного устройства в соответствии с таблицей трансляции (пример 12.3, пример 12.4). Это преобразование нужно осуществлять после выполнения маршрутизации.

Рис. 12.1. NAT для доступа в сеть Интернет

При публикации локального информационного ресурса в сети Интернет (рис. 12.2) используется трансляция адреса получателя (destination NAT). Обращения узлов Интернет к определенному порту Интернет-адреса граничного устройства транслируются в обращения к соответствующему узлу локальной сети (пример 12.5). Данное преобразование нужно осуществлять до выполнения маршрутизации.

Рис. 12.2. Публикация локального ресурса в сети Интернет с помощью NAT

Примечание: В примерах пример 12.3, пример 12.4, пример 12.5 на внешних интерфейсах указаны адреса из локального диапазона. Для доступа к сети Интернет следует реальные IP-адреса, из диапазона,

предоставляемом провайдером. В тоже время пример 12.3 можно трактовать как случай подключения шлюза локальной сети к шлюзу Интернет (например ADSL-модему), на котором есть Интернет-интерфейс и происходит еще одна трансляция сетевых адресов.

Рассмотрим маршрутизатор на основе Linux с двумя сетевыми интерфейсами [50]: интерфейс eth0 (IP-адрес 192.168.56.102/24, шлюз 192.168.56.1) используется для доступа к внешней сети, а интерфейс eth1 (IP-адрес 192.168.1.1/24) включен в локальную сеть (клиенты локальной сети должны иметь адреса из сети 192.168.1.0/24 и шлюз по умолчанию 192.168.1.1).

Пусть необходимо транслировать запросы локальных клиентов в адрес 192.168.56.102 при передаче их шлюзу 192.168.56.1. Для этого на маршрутизаторе необходимо включить пересылку пакетов, а затем добавить в таблицу *NAT* цепочки POSTROUTING правило маскировать (MASQUERADE) пакеты, попадающие после маршрутизации на интерфейс eth0 и имеющие исходный адрес из сети 192.168.1.0/24:

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

Например, выполнив с узла локальной сети команду `ping -c 1 192.168.56.1` и на маршрутизаторе проанализировав с помощью `tcpdump` сетевые пакеты, можно убедиться, что трансляция действительно происходит. Эхо-запрос и эхо-ответ на внутреннем интерфейсе маршутизатора:

```
22:31:56.822772 IP 192.168.1.2 > 192.168.56.1: ICMP echo request, id 4813
22:31:56.825035 IP 192.168.56.1 > 192.168.1.2: ICMP echo reply, id 48135,
```

Листинг 12.3. Настройка маршрутизатора с функцией NAT

На внешнем интерфейсе:

```
22:31:56.822873 IP 192.168.56.102 > 192.168.56.1: ICMP echo request, id 4
22:31:56.824968 IP 192.168.56.1 > 192.168.56.102: ICMP echo reply, id 481
```

Примечание: Действие MASQUERADE подразумевает автоматическое определение исходного адреса трансляции из текущих настроек интерфейса, что полезно при использовании подключений, IP-адрес на

которых устанавливается динамически.

Пусть у маршрутизатора из предыдущего примера есть дополнительный IP-адрес 192.168.56.103 и необходимо в него транслировать все пакеты, принадлежащие узлу локальной сети 192.168.1.2, а пакеты остальных узлов локальной сети - в адрес 192.168.56.102. Для явного задания исходного IP-адреса трансляции необходимо использовать действие SNAT с параметром `--to-source`:

```
# iptables -I POSTROUTING 1 -s 192.168.1.2 -o eth0 -j SNAT --to-source 1
# iptables -I POSTROUTING 2 -s 192.168.1.0/24 -o eth0 -j SNAT --to-sourc
```

Листинг 12.4. Указание IP-адреса для трансляции

Пусть необходимо, чтобы SSH-сервер на порту 8222 узла локальной сети 192.168.1.2 (пример 12.3) был доступен на внешнем адресе маршутизатора. Для этого необходимо использовать действие DNAT с параметром `--to-destination`:

```
# iptables -t nat -A PREROUTING -i eth1 -p tcp -d 192.168.56.102 --dport 8
  -j DNAT --to-destination 192.168.1.2
```

Листинг 12.5. Публикация SSH-сервера

Замечания по работе системы разрешения доменных имен

Система доменных имен Интернет (DNS, Domain Name System) представляет собой иерархическую распределенную систему сопоставления символьных имен узлов и сетевых служб их IP-адресам, а также маршрутизации электронной почты. Иерархия DNS отражена в многоуровневой структуре доменных имен (например: www.intuit.ru, ru — домен 1-го уровня, intuit — 2-го, www — 3-го).

Для разрешение IP-адреса по доменному имени используется рекурсивный поиск. Рекурсия начинается с обращения к одному из корневых DNS-серверов, содержащих ссылки на DNS-сервера, ответственные за домены 1-го уровня, далее отправляется запрос о расположении домена 2-го уровня к полномочному DNS-серверу

домена 1-го уровня и т. д. Рекурсивный DNS-сервер обычно в целях снижения трафика кэширует запросы (DNS-кэш).

Примечание: Использование символьного имени информационного ресурса в рамках сети Интернет требует внесения соответствующей записи в полномочный DNS-сервер домена (или зоны), к которому принадлежит данный ресурс.

Для полноценного взаимодействия с сетью Интернет на узлах локальной сети должен быть указан один или несколько IP-адресов рекурсивных DNS-серверов. В качестве таковых можно использовать DNS-сервера провайдера или настроить локальный DNS-кэш.

Примечание: Альтернативный вариант — на узле установить программное обеспечение, способное выполнять рекурсивные DNS-запросы.

Традиционно в Linux поведение клиентской подсистемы DNS определяется следующими файлами:

- /etc/resolv.conf, в котором могут указываться домен узла, домен подстановки при разрешении неполных имен и список DNS-серверов;
- /etc/hosts — содержит статическую таблицу разрешения имен узлов;
- /etc/host.conf — содержит некоторые настройки подсистемы DNS.

В некоторых случаях для динамического изменения настроек DNS в зависимости от активных внешних подключений, используется пакет resolvconf. NetworkManager также может менять содержимое файла resolv.conf в соответствие с производимыми пользователем изменениями. В Debian также на содержимое файла resolv.conf могут влиять записи о DNS-серверах в файле /etc/network/interfaces. Для тестирования работы подсистемы DNS могут использоваться утилиты `nslookup` или `dig` (в Debian содержатся в пакете `dnsutils`).

Пример 12.6. .

```
# dig -b 192.168.200.2 @192.168.200.1 www.intuit.ru a
```

```
; <<>> DiG 9.6-ESV-R3 <<>> -b 192.168.200.2 @192.168.200.1 www.in
; (1 server found)
;; global options: +cmd
;; Got answer:
;; - >>HEADER<< - opcode: QUERY, status: NOERROR, id: 4921
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 4
;; QUESTION SECTION:
;www.intuit.ru. IN A
;; ANSWER SECTION:
www.intuit.ru. 1295 IN A 194.67.246.18
;; AUTHORITY SECTION:
intuit.ru. 1295 IN NS ns.intuit.ru.
intuit.ru. 1295 IN NS ns2.osp.ru.
intuit.ru. 1295 IN NS ns1.intuit.ru.
intuit.ru. 1295 IN NS ns3.osp.ru.
;; ADDITIONAL SECTION:
ns.intuit.ru. 3447 IN A 194.67.165.74
ns1.intuit.ru. 3447 IN A 194.67.246.5
ns2.osp.ru. 3447 IN A 81.13.33.67
ns3.osp.ru. 3447 IN A 194.67.168.66
;; Query time: 18 msec
;; SERVER: 192.168.200.1#53(192.168.200.1)
;; WHEN: Sun May 29 19:26:16 2011
;; MSG SIZE rcvd: 186
```

Утилита `dig` полезна для тестирования DNS в сложных конфигурациях. В данном случае ключ `-b` позволяет указать исходный IP-адрес, после `@` указывается IP-адрес сервера, к которому делается запрос, а затем — имя, которое нужно разрешить и тип записи (`a`).

Пример настройки сервера доступа в Интернет для локальной сети

Пусть есть подключение к инфраструктуре провайдера по технологии Ethernet, а Интернет-доступ осуществляется с помощью технологии PPTP VPN. Реализуем в VirtualBox на основе Linux схему, состоящую из сервера доступа (первая виртуальная машина) к сети Интернет для виртуального сетевого сегмента, представленного еще одной

виртуальной машиной - клиентом. Для доступа сервера к Интернет будет использоваться подключение VirtualBox "Сетевой мост", виртуальный локальный сегмент будет использовать "Внутреннюю сеть". Настроим на сервере для клиентов виртуальной локальной сети DNS-кэш на основе программы dnsmasq [50] таким образом, чтобы клиенты отправляли DNS-запросы на локальный адрес сервера, а сервер обращался к DNS-серверу провайдера. Дополнительно также будет настроен DHCP-сервер на локальном интерфейсе.

1. Запустим сервер, предварительно настроив два сетевых адаптера сервера на соответствующие виртуальные подключения. Пусть сетевой адаптер сервера eth0 подключен к сетевому мосту, а eth1 — к внутренней сети. Настроим eth0 на динамическое получение параметров IP (в сети провайдера), а на eth1 установим статический IP-адрес 192.168.1.1/24. В Debian для этого надо отредактировать файл /etc/network/interfaces и перезапустить сетевую подсистему:

```
# cat /etc/network/interfaces
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto eth1
iface eth1 inet static
   address 192.168.1.1
   netmask 255.255.255.0
```

2. Далее установим на сервере пакет *pptp*-linux с помощью команды *apt*-get:

```
# apt-get install pptp-linux
```

В файле /etc/ppp/chap-secrets укажем логин, имя подключения и пароль для подключения к сети провайдера:

```
# cat /etc/ppp/chap-secrets
```

```
user0013 vpn Sdnf5r1
```

Для инициализации pptp-туннеля и настройки маршрутизации на сервере напишем скрипт upptplink.sh:

```
# cat upserver.sh
#!/bin/sh

# адрес VPN-сервера
SERVER ="10.10.10.10"
# имя пользователя
USER=user0013
# имя ppp-интерфейса
PIF=ppp0

pppd file /etc/ppp/options.pptp defaultroute replacedefaultroute name $US
  remotename vpn pty  "pptp $SERVER —nolaunchpppd"

sleep 5
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o $PIF -s 192.168.1.0/24 -j MASQ
```

Утилита *pppd* устанавливает соединение точка-точка с VPN-сервером с помощью утилиты *pptp*. Основные опции *pppd* указаны в файле /etc/ppp/options.pptp (в данном случае используются настройки по умолчанию), в командной строке указаны опции `defaultroute` и `replacedefaultroute` для использования туннеля в качестве шлюза по умолчанию и перезаписи, если необходимо, текущего шлюза по умолчанию, опция `name` — имя пользователя, `remotename` — имя vpn-подключения.

В скрипте также включается функция пересылки и создается правило для трансляции запросов узлов локальной сети в адрес, установленный на ppp-интерфейсе.

3. Для Debian `dnsmasq` можно установить из репозитория (пакет называется dnsmasq-base):

```
# apt-get install dnsmasq-base
```

Далее создадим файл конфигурации /etc/dnsmasq.conf:

```
# cat /etc/dnsmasq.conf
listen-address=192.168.1.1
no-dhcp-interface=eth0,ppp0
no-resolv
server=10.10.10.10
dhcp-range=192.168.1.10,192.168.1.253,24h
cache-size=300
dhcp-leasefile=/var/lib/misc/dnsmasq.leases
```

Опция `listen-address` указывает на каком интерфейсе слушать клиентские dns-запросы, `no-resolv` — не использовать файл /etc/resolv.conf для настройки родительского DNS-сервера, `server` — указать DNS-сервер провайдера.

Примечание: Кроме DNS-форвардинга в dnsmasq реализована функция выдачи IP-адресов клиентам по протоколу DHCP (DHCP-сервер). Соответственно опция `dhcp-range` — диапазон динамически выдаваемых адресов, `no-dhcp-interface` — запретить на указанных интерфейсах выдавать IP-адреса. Необходимые настройки протокола IP для передачи DHCP-клиентам `dnsmasq` сформирует автоматически (маску подсети, шлюз по умолчанию, адрес DNS-сервера).

4. Запустим еще одну виртуальную машину (клиента), предварительно подключив ее сетевой адаптер к внутренней сети. В системе сетевой интерфейс настроим на динамическую настройку IP-параметров:

```
# cat /etc/network/interfaces | grep eth0
auto eth0
iface eth0 inet dhcp
```

После инициализации сетевой подсистемы на клиенте можно проверить доступ в сети Интернет.

Ключевые термины

Виртуальные частные сети (VPN, Virtual Private Network) - обобщённое название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети.

xDSL - семейство технологий, использующих абонентские линии DSL (Digital Subscriber Line), "x" служит обозначением конкретной технологии.

Трансляция сетевых адресов (NAT, Network Address Translation) - механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса пакетов.

PPPoE (Point-to-point protocol over Ethernet) — сетевой протокол канального уровня передачи кадров PPP через Ethernet. В основном используется xDSL-сервисами. Предоставляет дополнительные возможности (аутентификация, сжатие данных, шифрование).

3G (технологии мобильной связи 3 поколения) — набор услуг, который объединяет как высокоскоростной мобильный доступ с услугами сети Интернет, так и технологию радиосвязи.

PPTP (Point-to-Point Tunneling Protocol) — туннельный протокол типа точка-точка, позволяющий компьютеру устанавливать защищённое соединение с сервером за счёт создания специального туннеля в стандартной, незащищённой сети. PPTP помещает (инкапсулирует) кадры PPP в IP-пакеты для передачи IP-сети.

Краткие итоги

1. Обычно для доступа к сети Интернет клиенту необходимо устанавливать специальное оборудование, а также настраивать логические подключения с использованием технологий VPN. Широко распространенными технологиями доступа в Интернтет являются: xDSL, 3G, Wi-Fi, WiMax, Ethernet и т.д.
2. Для сегмента локальной сети доступ в Интернет может быть организован с помощью граничного устройства с включенной функцией трансляции адреса источника: при взаимодействии с узлом Интернет адрес локального узла подменяется Интернет-адресом граничного устройства в соответствии с таблицей трансляции. Альтернативным вариантом является установка

прикладного прокси-сервера.

3. Для публикации локального информационного ресурса в сети Интернет может использоваться трансляция адреса получателя на граничном устройстве: обращения узлов Интернет к определенному порту внешнего адреса граничного устройства преобразуются в обращения к данному узлу локальной сети.
4. Важным элементом во взаимодействии узлов Интернет является система DNS. Для корректного определения IP-адресов по символьным именам на узле должен быть настроен адрес DNS-сервера.

Упражнения

1. С помощью двух виртуальных машин в среде VirtualBox реализовать пример 12.1
2. С помощью двух виртуальных машин и основной системы в среде VirtualBox реализовать пример 12.3, пример 12.4. Настроить на шлюзе DHCP-сервер для динамической выдачи IP-адресов узлам локальной сети. Разрешить узлам локальной сети выход во внешнюю сеть 10.0.0.0/8 и доступ к шлюзу, а прочий трафик запретить.

Список литературы

1. Костромин В.А, Основы работы в ОС Linux , URL: http://ituit.ru , Интернет-университет ИТ, 2006

2. Debian GNU/Linux, URL: http://debian.org

3. Курячий Г.В., Масличный К.А, Операционная система Linux , URL: http://ituit.ru , Интернет-университет ИТ, 2005

4. Debian Reference Osamu Aoki, URL: http://www.debian.org/doc/manuals/debian-reference/

5. VirtualBox, URL: http://virtualbox.org

6. Oracle VM VirtualBox User Manual, URL: http://www.virtualbox.org/manual/UserManual.html

7. Олифер Н., Олифер В, Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. Санкт-Петербург: , Питер, 2010. – 944 с

8. Linux kernel map, URL: http://www.makelinux.net/kernel_map

9. М. Тим Джонс, Анатомия сетевого стека в Linux От сокетов до драйверов устройств, URL: http://www.ibm.com/developerworks/ru/library/l-linux-networking-stack/

10. Netfilter project, URL: http://www.netfilter.org/

11. net-tools, URL: http://www.linuxfoundation.org/collaborate/workgroups/networking/net-tools

12. iproute2, URL: http://www.linuxfoundation.org/collaborate/workgroups/networking/iproute2

13. Network Manager , URL: http://projects.gnome.org/NetworkManager/

14. Wicd homepage, URL: http://wicd.sourceforge.net/

15. UncomplicatedFirewall, URL: https://wiki.ubuntu.com/UncomplicatedFirewall

16. Shoreline Firewall, URL: http://shorewall.net/

17. Firewall Builder, URL: http://www.fwbuilder.org/

18. FireStarter, URL: http://sourceforge.net/projects/firestarter/

19. Mike Muuss, The Story of the PING Program , URL: http://ftp.arl.mil/~mike/ping.html

20. Программа сетевой академии Cisco CCNA 1 и 2. Вспомогательное руководство, 3-е изд., с исп, М.:Издательский дом "Вильямс", 2005. – 1168 с. (стр. 303-352)

21. Guide to IP Layer Network Administration with Linux, URL: http://linux-ip.net/html/index.html

22. Берлин А.Н, Основные протоколы интернет. 4 Лекция. Протокол Интернет версии 4 , URL: http://www.intuit.ru/department/network/internetprot/4/internetprot_4.html , Интернет-университет Информационных технологий

23. IP Calculator, URL: http://jodies.de/ipcalc

24. IPROUTE2 Utility Suite Howto, URL: http://www.policyrouting.org/iproute2.doc.html

25. Guide to IP Layer Network Administration with Linux, URL: http://linux-ip.net/html/index.html

26. Linux Advanced Routing & Traffic Control HOWTO , URL: оригинал: http://lartc.org/howto/, перевод: http://www.opennet.ru/docs/RUS/LARTC/

27. Osamu Aoki Debian Reference, URL: http://www.debian.org/doc/manuals/debian-reference

28. VLAN на xgu.ru , URL: http://xgu.ru/wiki/VLAN

29. The TCP/IP Guide, URL: http://www.tcpipguide.com/free/index.htm
30. About iptraf, URL: http://iptraf.seul.org/about.html
31. Анализ функционирования сети, URL: http://www.opennet.ru/prog/sml/137.shtml
32. Учет трафика , URL: http://www.opennet.ru/prog/sml/47.shtml
33. Iperf, URL: http://iperf.sourceforge.net/
34. Русскоязычная документация Wiki, URL: http://help.ubuntu.ru/wiki/ssh
35. OpenSSH, URL: http://www.openssh.com/
36. Free SFTP, FTP and SCP client for Windows , URL: http://winscp.net/eng/docs/lang:ru
37. PuTTY: Free Telnet/SSH Client, URL: http://www.chiark.greenend.org.uk/~sgtatham/putty/
38. Apache HTTP Server, URL: http://httpd.apache.org/
39. Apache HTTP Server Documentation , URL: http://httpd.apache.org/docs/
40. Squid: Optimising Web Delivery, URL: http://www.squid-cache.org/
41. Wireshark, URL: http://www.wireshark.org
42. Tcpdump&Libpcap, URL: http://www.tcpdump.org/
43. Policy Routing With Linux - Online Edition, URL: http://www.policyrouting.org/PolicyRoutingBook/ONLINE/TOC.html
44. Matthew G. Marsh, The netfilter.org project, URL: http://www.netfilter.org/
45. Страница из Wikipedia, URL: http://ru.wikipedia.org/wiki/Netfilter
46. Руководство по iptables (Iptables Tutorial 1.1.19), URL: http://www.opennet.ru/docs/RUS/iptables/
47. Oskar Andreasson, Программа сетевой академии Cisco CCNA 1 и 2. Вспомогательное руководство, 3-е изд., с испр
48. Настраиваем роутер на Linux Ubuntu NAT + DHCP + Squid, URL: http://mannix.ru/ubuntu/nastraivaem-router-na-linux-ubuntu-nat-dhcp-squid.html , М.:Издательский дом "Вильямс", 2005. – 1168 с
49. Squid: Optimising Web Delivery, URL: http://www.squid-cache.org/
50. Dnsmasq, URL: http://www.thekelleys.org.uk/dnsmasq/doc

Содержание