Впервые опубликовано на developerWorks 08.11.2005

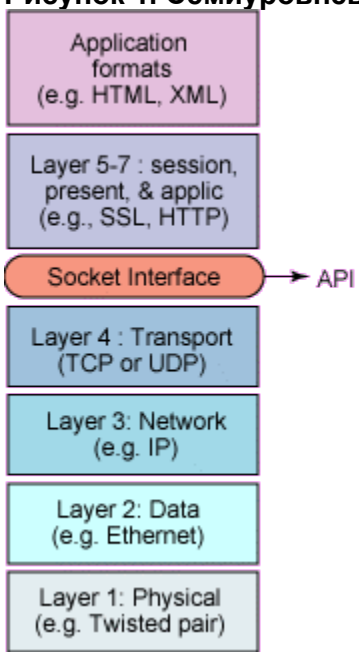
Подготовка к сдаче экзамена LPI: Конфигурирование сети

Intermediate Level Administration (LPIC-2) тема 205

О настройке сети

При обсуждении сетей в Linux и настройке сети следует держать в умесемиуровневую модель OSI:

Рисунок 1. Семиуровневая модель OSI



То, что мы называем "настройкой сети" на самом деле затрагивает настройку параметров второго и третьего уровней -- канального и сетевого -- а также интерфейсов между ними. На практике это включает в себя настройку Ethernet или последовательных интерфейсов наподобие модема для канального уровня, и настройку протокола Internet Protocol (IP) для сетевого уровня. Последующие руководства этой серии рассматривают более высокие уровни организации, хотя большинство серверных приложений, обсуждаемых здесь, не различают четко все семь уровней (или даже четыре верхних уровня, на которых они работают).

Первый уровень это физический уровень, провода (или же беспроводные каналы) и связи между ними. Реальному сетевому администратору необходимо быть готовым к исследованию кабелей и установке новой сетевой периферии время от времени (однако эти вопросы выходят за рамки данных руководств). Ясно, что плохой провод, сетевая карта или сломанный коннектор могут создать проблемы в сети, также как и направильно настроенное программное обеспечение.

Четвертый уровень это транспортный уровень; а именно TCP или UDP в IP сетях. TCP и UDP используются на верхних уровнях посредством Berkeley Sockets Interface, хорошо оттестированной библиотеке, встречаемой на всех современных компьютерных системах. О том, как приложения (как те, о которых рассказано далее в этой серии руководств) используют TCP или UDP смотрите руководства "Программирование сокетов в Linux, Часть 1" и "Программирование сокетов в Linux, Часть 2."

Другие источники

Как и для большинства инструментов Linux, man-страницы содержат ценную информацию.За более подробной информацией можно обратиться в проект Linux Documentation Project, содержащий много полезных документов, особенно HOWTO. Множество книг по сетям Linux может быть особенно полезным, как например книга издательства O'Reilly под названиемСетевое администрирование TCP/IPавтора Крэйга Ханта. Вы найдете эти и другие источники в разделеРесурсыэтого руководства.

Настройка сети

Протокол разрешения адреса

Первое, что необходимо помнить об устройствах Ethernet -- как беспроводных 802.11a/b/g, так и более традиционных сетевых карт CAT5/CAT6 -- это то, что у каждого Ethernet устройства имеется уникальный 6 байтный идентификатор. Эти идентификаторы распределены по группам, каждая из которых присвоена производителю; вы можете посмотреть эти группы в IANA. Ethernet в общем "просто работает" на физическом уровне, однако системе требуется отобразить идентификатор Ethernet на используемый IP адрес, чтобы была возможность работы с IP.

Протокол Address Resolution Protocol (ARP) позволяет машинам узнавать IP адреса друг друга внутри локальной Ethernet сети. Что касается протокола, ARP в основном реализован в драйвере сетевого устройства (как модуль ядра); инструмент arp позволяет вам посмотреть статус системы ARP и немного ее настроить. В данный момент мы предположим, что у каждой машины есть свой IP адрес, либо статический либо полученный с помощью DHCP.

Когда система Linux (или любое другое устройство Ethernet) желает обратиться к IP адресу, то с помощью широковещательного запроса Ethernet ARP посылает сообщение с запросом "кто есть X.X.X.X сообщите Y.Y.Y.Y". Целевая система формирует ARP ответ "X.X.X.X это hh:hh:hh:hh:hh:hh" и посылает его запрашивающему устройству. Ответ ARP кэшируется короткое время в /proc/net/arp, чтобы избежать постоянного восстановления отображения между аппаратными Ethernet адресами и IP адресами.

Утилита arp

Утилита Linux arp позволяет вам изучать и модифицировать статус ARP отображений. Простейший доклад о статусе может выглядеть как в Листинге 1:

Листинг 1. Доклад о статусе ARP

```
$ arp -n
Address          HWtype  HWaddress           Flags Mask    Iface
192.168.2.1      ether   00:03:2F:09:61:C7   C             eth0
```

Здесь говорится, что определенному устройству назначен в этой сети адрес 192.168.2.1 (судя по виду, этот адрес соответствует маршрутизатору\шлюзом, что в данном случае так и есть). Тот факт, что только одна запись содержится в этом списке, не означает, что в сети больше не существует других устройств, так как записи ARP других устройств могут быть просрочены. ARP стирает записи после короткого промежутка времени -- в течение нескольких минут, а не секунд или часов -- чтобы позволить сетям самим реконфигурироваться в случае добавления или удаления устройств или изменения установок на машинах. Кэшируя запись ARP в течение короткого времени, можно не посылать новые запросы во время работы большинства сетевых сессий.

Любой вид IP запроса машины, которая может находиться в локальной сети, заставляет ядро посылать ARP запрос; если получен ответ ARP , то машина добавляется в кэш ARP (как в Листинге 2):

Листинг 2. Взаимодействие с другими IP адресами

```
$ ping -c 1 192.168.2.101 > /dev/null
$ ping -c 1 192.168.2.101 > /dev/null
$ ping -c 1 192.168.2.102 > /dev/null
$ ping -c 1 192.168.32.32 > /dev/null
$ ping -c 1 192.168.32.32 > /dev/null
$ arp -n
Address          HWtype  HWaddress           Flags Mask    Iface
192.168.2.1      ether   00:03:2F:09:61:C7   C             eth0
192.168.2.101    ether   00:30:65:2C:01:11   C             eth0
192.168.2.100    ether   00:11:24:9D:1E:4B   C             eth0
192.168.2.102    ether   00:48:54:83:82:AD   C             eth0
```

В этом случае, первые четыре адреса действительно существуют в сети Ethernet, но 192.168.32.32 не существует, поэтому от него ARP ответ не получен. Заметим, что если вам удалось подключиться к адресам не через локальный маршрут, то в кэш ARP ничего не добавится (смотри Листинг 3):

Листинг 3. Ничего не добавляется в кэш ARP

```
$ ping -c 1 google.com
PING google.com (216.239.57.99) 56(84) bytes of data.
64 bytes from 216.239.57.99: icmp_seq=1 ttl=235 time=109 ms
--- google.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 109.123/109.123/109.123/0.000 ms
$ arp -n
Address          HWtype  HWaddress           Flags Mask    Iface
192.168.2.1      ether   00:03:2F:09:61:C7   C             eth0
```

Google доступен (так как маршрут уже настроен), но 216.239.57.99 не локален, поэтому в ARP ничего не добавляется. Седьмое руководство в этой серии, по теме 214, рассматривает проблемы с сетью и демонстрирует, как установить значение ARP вручную.

PPP, PAP и CHAP

Протокол Point-to-Point Protocol (PPP) используется для установления связи с Internet через dial-up модемы, прямые последовательные соединения, DSL, и другие типы связи точка-точка (иногда включая PPPoE как "псевдо-уровень" поверх Ethernet, этот протокол обеспечивает установление подключения). Демон pppd работает совместно с включенным в ядро PPP драйвером, чтобы установить и поддерживать PPP соединение с другой системой (обычно называемой узлом) и установить адреса Internet Protocol (IP) для каждого конца соединения.

PPP, в особенности pppd, запрашивает аутентификацию у второго участника соединния и предоставляет ему свои аутентификационные данные. Такая аутентификация выполняется с использованием простой системы паролей Password Authentication Protocol (PAP) или сессионной системы Challenge Handshake Authentication Protocol (CHAP). Из двух упомянутых, CHAP более безопасен, если обе стороны его поддерживают.

Опции PPP, как правило, хранятся в /etc/ppp/options. Конфигурация PAP осуществляется через файл паролей PAP /etc/ppp/pap-secrets, для CHAP она осуществляется через файл паролей CHAP /etc/ppp/chap-secrets.

Файл паролей PAP/CHAP

Файл /etc/ppp/pap-secrets содержит разделенные пробелами поля клиента, сервера, пароля, и допустимого локального IP адреса. Последнее поле может быть пустым (и как правило оно именно пустое при динамическом назначении IP адреса). Файл паролей PAP должен быть сконфигурирован отдельно для каждого пользователя.

Хотя PPP -- это протокол взаимодействия равноправных систем, в целях подключения мы будем называть запрашивающую машину клиентом, а ожидающую машину сервером. Например, машина bacchus в моей сети может иметь следующий файл настроек:

Листинг 4. Настройка pap-secrets на bacchus

```
# Every regular user can use PPP and uses passwords from /etc/passwd
# INBOUND connections
# client   server  secret            acceptable local IP addresses
*          bacchus ""                *
chaos      bacchus chaos-password
# OUTBOUND connections
bacchus      *     bacchus-password
```

Машина bacchus будет принимать соединения от любых обычных пользователей, а также принимать соединения с машины chaos (требуя пароль chaos-password в последнем случае). При подключении к другим машинам bacchus будет просто использовать свое собственное имя и предлагать пароль bacchus-password каждому узлу.

Соответственно машина chaos в моей сети может содержать следующий файл:

Листинг 5. Машина chaos более консервативна в выборе соединений

```
# client   server  secret            acceptable local IP addresses
chaos      bacchus chaos-password
bacchus    chaos   bacchus-password
```

Машина chaos более консервативна в отношении к кому она будет подключаться. Она обменялась параметрами доступа только с bacchus. Вы можете настроить каждый файл /etc/ppp/options и определить имя пользователя и пароль, если требуется.

Использование паролей CHAP требует, чтобы оба узла могли аутентифицировать друг друга. При условии, что двустороння аутентификация настроена в паролях PAP, файл паролей CHAP может выглядеть, как в приведенных выше примерах.

Соединение с помощью mgetty

Файл паролей PAP может быть использован с функцией AUTO_PPP mgetty. mgetty 0.99+ уже настроена на запуск pppd с опцией login. Она сообщает, pppd надо обратиться за справкой к /etc/passwd (и /etc/shadow в свою очередь) после того как пользователь передал этот файл.

В общем, программа getty может быть настроена, чтобы принимать соединения от последовательных устройств, включая модемы и последовательные порты. Например, для проводной линии или консоли tty, вы можете запустить:

```
/sbin/getty 9600 ttyS1
```

в вашем терминале. Для старых телефонных линий с модемом в 9600/2400/1200 бод можно запустить команду:

```
/sbin/getty -mt60 ttyS1 9600,2400,1200.
```

Настройка маршрутизации

В разделе обсуждения протокола Address Resolution Protocol мы видели, как назначаются адреса в локальной сети. Однако чтобы взаимодействовать с машинами вне локальной сети, необходимо иметьмаршрутизатор. В общих чертах маршрутизатор -- это просто компьютер, который подсоединяется к нескольким сетям, и поэтому может брать пакеты из одной сети и передавать их в другие. Именно отсюда и пошло название "Internet": это "сеть, состоящая из сетей", в которой каждый маршрутизатор, в конечном счете, может достучаться до любой другой сети, которая "подключена к Internet."

Пятое руководство этой серии по теме 210 рассматривает управление клиентом сети и DHCP. DHCP назначит как IP адреса, так и адрес маршрутизатора. Однако, если у клиента фиксированный IP адрес, или в целях тестирования, команда Linux route позволит вам просмотреть и модифицировать таблицы маршрутизации. Более новая команда ip также позволит вам модифицировать таблицы маршрутизации, используя более мощный синтаксис.

Таблица маршрутизации просто позволяет вам определить, через какой маршрутизатор или узел посылать пакет, на основе определенного шаблона в адресе. Шаблон адреса определяется комбинацией адреса и маски подсети. Маска подсети -- это битовый шаблон, обычно представляется в форме групп чисел, разделенных точками, которые сообщают ядру о том, какие биты адреса доставки считать как сетевой адрес, а какие оставшиеся биты считать как подсеть. Команда ip может принять упрощенный /NN формат битовых масок. В общем, в маске и адресе нулевые биты это "метасимволы".

Например, простая сеть с одним внешним шлюзом может содержать таблицу маршрутизации как в Листинге 6:

Листинг 6. Типичная простая таблица маршрутизации

```
$ route -n
Kernel IP routing table
Destination  Gateway      Genmask         Flags Metric Ref Use Iface
192.168.2.0  0.0.0.0      255.255.255.0   U     0      0     0 eth0
0.0.0.0      192.168.2.1  0.0.0.0         UG    0      0     0 eth0
```

Это значит, что сетевой пакет на любой IP адрес, который соответствует шаблону "192.168.2.*" преназначен компьютеру из локальной сети и будет направлен прямо на нужный узел (полученный при помощи ARP). Все остальные пакеты будут посылаться на маршрутизатор "192.168.2.1", который перенаправит их по назначению. Машина 192.168.2.1 должна быть подсоединена к одной или нескольким внешним сетям.

Однако в более сложном случае вы можете по другому определить шаблон назначения. Придумаем пример, положим, что вы хотите направить определенные

адреса /16 через другие шлюзы. Вы можете сделать это, как показано в Листинге 7:

Листинг 7. Изменение маршрута сетей /16

```
$ route add -net 216.109.0.0 netmask 255.255.0.0 gw 192.168.2.2
$ route add -net 216.239.0.0 netmask 255.255.0.0 gw 192.168.2.3
$ route -n
Kernel IP routing table
Destination  Gateway      Genmask        Flags Metric Ref Use Iface
192.168.2.0  0.0.0.0      255.255.255.0  U     0      0     0 eth0
216.109.0.0  192.168.2.2  255.255.0.0    UG    0      0     0 eth0
216.239.0.0  192.168.2.3  255.255.0.0    UG    0      0     0 eth0
0.0.0.0      192.168.2.1  0.0.0.0        UG    0      0     0 eth0
```

Пакеты на адреса вида "216.109.*" и "216.239.*" будут теперь направляться через маршрутизаторы 192.168.2.2 и 192.168.2.3, соответственно (оба находятся в локальной сети). Пакеты на локальные адреса или адреса, несоответствующие шаблону, будут направляться так же, как и раньше. Вы можете использовать командуroute delete, чтобы удалить маршруты.

Сложная настройка сети и разрешение проблем

О сетевых утилитах

Linux поставляется с набором стандартных утилит, которые вы можете использовать для настройки и разрешения проблем с сетью. Хотя большая часть сетевого кода Linux находится в самом ядре, почти все, что касается поведения сети, можно настроить с помощью утилит командной строки. Многие дистрибутивы поставляются с инструментами более высокого уровня, возможно графическими. Но они могут сделать туже самую работу, что и инструменты командной строки.

Утилита ping

Самый простой способ проверить, имеет ли узел с Linux доступ к IP адресу (или К именованному узлу, в случае настроенных DNS и/или /etc/hosts) состоит в использовании утилиты ping. ping работает на уровне IP и не полагается на канальный уровень как TCP или UDP. ping вместо них использует протокол Internet Control Message Protocol (ICMP). Если вы не можете достичь узла с помощью ping, то почти наверняка вы не сможете с ним связаться и с помощью других инструментов, поэтому ping всегда является первым шагом в установлении возможности подключения к узлу (man ping может предоставить описание параметров команды).

По умолчанию, ping посылает сообщение каждые две секунды до тех пор, пока не будет прервана ее работа, однако вы можете изменить время, ограничить число сообщений и подробности вывода. Во время работы ping выводит время

путешествия пакета и число потерянных пакетов, но в большинстве случаев вы либо сможете получить ответ от узла, либо нет.В Листинге 8 приведены некоторые примеры:

Листинг 8. Работа ping с локальными и нелокальными узлами

```
$ ping -c 2 -i 2 google.com
PING google.com (216.239.37.99): 56 data bytes
64 bytes from 216.239.37.99: icmp_seq=0 ttl=237 time=43.861 ms
64 bytes from 216.239.37.99: icmp_seq=1 ttl=237 time=36.956 ms

--- google.com ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 36.956/40.408/43.861 ms

$ ping 192.168.2.102
PING 192.168.2.102 (192.168.2.102): 56 data bytes
64 bytes from 192.168.2.102: icmp_seq=0 ttl=255 time=4.64 ms
64 bytes from 192.168.2.102: icmp_seq=1 ttl=255 time=2.176 ms
^C
--- 192.168.2.102 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 2.176/3.408/4.64 ms
```

Утилита iconfig

Сетевые интерфейсы настраиваются с помощью инструмента ifconfig. Обычно она запускается как часть процесса инициализации, но в некоторых случаях интерфейсы могут быть модифицированы и настроены позже (особенно при отладке). Если вы запустите ifconfig без ключей, то увидите отображение текущего сетевого статуса. Вы можете использовать ifconfig <interface> up и ifconfig <interface> down, чтобы запустить и остановить сетевые интерфейсы. Некоторые ключи изменяют формат отображения или ограничивают вывод только для конкретных интерфейсов. В man ifconfig можно узнать подробности.

Дополнительная информация может выглядеть как в Листинге 9:

Листинг 9. Использование ifconfig для просмотра информации о сетевых интерфейсах

```
$ ifconfig
eth0  Link encap:Ethernet  HWaddr 00:12:F0:21:4C:F8
      inet addr:192.168.2.103  Bcast:192.168.2.255  Mask:255.255.255.0
      inet6 addr: fe80::212:f0ff:fe21:4cf8/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:540 errors:0 dropped:0 overruns:0 frame:0
      TX packets:233 errors:0 dropped:0 overruns:0 carrier:1
      collisions:0 txqueuelen:1000
      RX bytes:49600 (48.4 KiB)  TX bytes:42067 (41.0 KiB)
      Interrupt:21 Base address:0xc000 Memory:ffcfe000-ffcfefff

ppp0  Link encap:Point-Point Protocol
      inet addr:10.144.153.104  P-t-P:10.144.153.51 Mask:255.255.255.0
      UP POINTOPOINT RUNNING  MTU:552  Metric:1
```

```
      RX packets:0 errors:0 dropped:0 overruns:0
      TX packets:0 errors:0 dropped:0 overruns:0


lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:4043 errors:0 dropped:0 overruns:0 frame:0
      TX packets:4043 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:368044 (359.4 KiB)  TX bytes:368044 (359.4 KiB)
```

В этом листинге видно, что настроено две сети, одна Ethernet и одна PPP (и еще присутствует повсеместный локальный интерфейс loopback). В других случаях у вас может быть настроено несколько интерфейсов Ethernet или других типов интерфейсов. Если это так, то говорят, что системасильно связана.

Утилита netstat

Утилиты Linux могут иметь схожий функционал. Инструмент netstat отображает информацию, которую также можно получить у нескольких утилит, как ifconfig и route. Вы также можете узнать общую расширенную статистику о сетевой активности. Например:

Листинг 10. Отчет сетевой статистики

```
$ netstat -s
Ip:
    12317 total packets received
    0 forwarded
    0 incoming packets discarded
    12255 incoming packets delivered
    11978 requests sent out
Icmp:
    1 ICMP messages received
    0 input ICMP message failed.
    ICMP input histogram:
       echo replies: 1
    0 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
Tcp:
    7 active connections openings
    5 passive connection openings
    0 failed connection attempts
    0 connection resets received
    3 connections established
    11987 segments received
    11885 segments send out
```

```
    0 segments retransmitted
    0 bad segments received.
    3 resets sent
Udp:
    101 packets received
    0 packets to unknown port received.
    0 packet receive errors
    92 packets sent
TcpExt:
    1 TCP sockets finished time wait in fast timer
    1490 delayed acks sent
    Quick ack mode was activated 5 times
    3632 packets directly queued to recvmsg prequeue.
    126114 of bytes directly received from backlog
    161977 of bytes directly received from prequeue
    1751 packet headers predicted
    3469 packets header predicted and directly queued to user
    17 acknowledgments not containing data received
    4696 predicted acknowledgments
    0 TCP data loss events
```

Другие утилиты

Есть также другие утилиты, о которых вы должны знать при настройке сети. Как обычно соответствующие им man-страницы содержат полную информацию по их использованию. Подробно они обсуждаются в седьмом руководстве по теме 214, которая рассматривает вопросы разрешения проблем, этой серии руководств.

tcpdump позволяет отслеживать все пакеты, которые проходят через сетевые интерфейсы, опционально можно ограничиться определенными интерфейсами или произвести фильтрацию по различным критериям. Часто такой отчет, который потом обрабатывается текстовыми утилитами, полезен при диагностике проблемы сети. Например, вы можете исследовать пакеты, которые приходят от конкретного удаленного узла.

Lsof выводит список всех открытых файлов в работающей Linux системе. Но в частности, вы можете использовать опцию lsof -i, чтобы просмотреть только псевдофайлы на наличие определенного IP соединния и вообще все сетевые соединения. Например:

Листинг 11. Использование lsof для просмотра псевдофайлов на наличие соединений

```
$ lsof -i
     COMMAND    PID USER   FD   TYPE DEVICE SIZE NODE
           NAME
     vino-serv 7812  dqm   33u  IPv4  12824
           TCP *:5900 (LISTEN)
     gnome-cup 7832  dqm   18u  IPv4  12865
           TCP localhost.localdomain:32771->localhost.localdomain:ipp
(ESTABLISHED)
     telnet    8909  dqm    3u  IPv4  15771
```

```
TCP 192.168.2.103:32777->192.168.2.102:telnet (ESTABLISHED)
```

nc и **netcat** -- это псевдонимы. netcat это простая утилита UNIX, которая читает и пишет данные по сети, используя протокол TCP или UDP. Это "back-end" инструмент, который можно использовать в других программах или скриптах. Во многих отношениях netcat похожа на telnet, но более гибка в плане работы с UDP и пересылке двоичных данных.