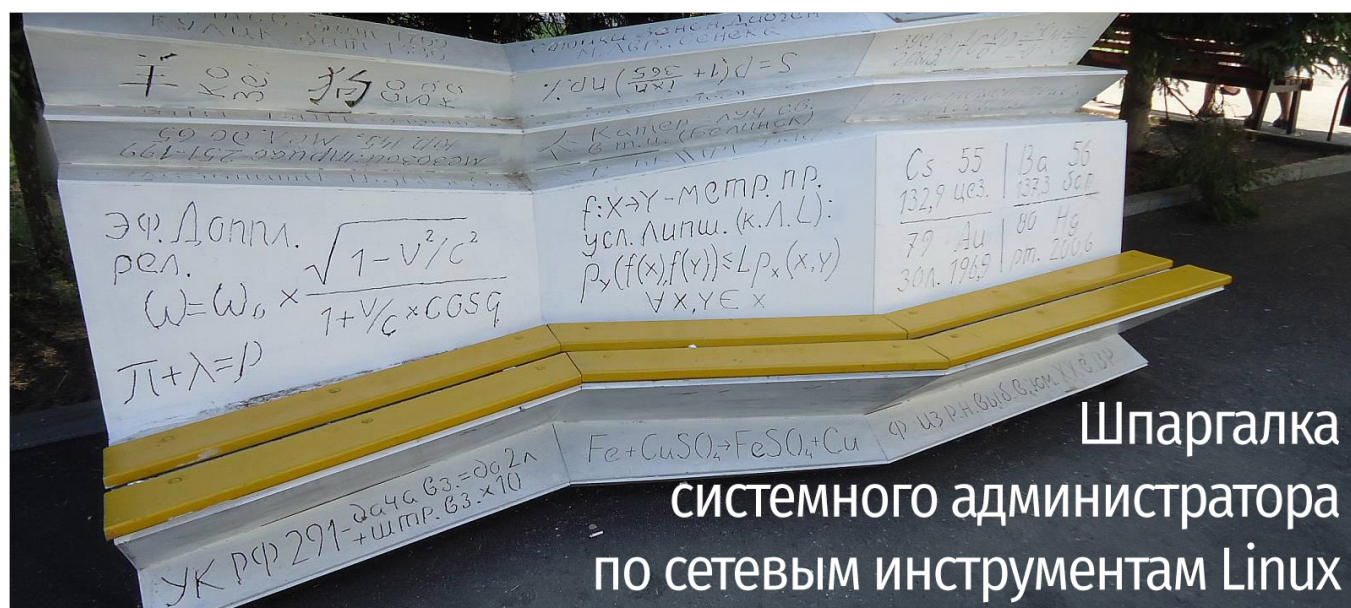


Шпаргалка системного администратора по сетевым инструментам Linux

June 05, 2019

В повседневные задачи системных администраторов входит работа с сетями и с подключённым к ним оборудованием. Нередко роль рабочего места администратора играет компьютер, на котором установлен какой-нибудь дистрибутив Linux. Утилиты и команды Linux, о которых пойдёт речь в материале, перевод которого мы публикуем сегодня, включают в себя список инструментов различной сложности — от простых, до продвинутых, которые предназначены для решения широкого спектра задач по управлению сетями и по диагностике сетевых неполадок.



В некоторых из рассматриваемых здесь примеров вы столкнётесь с сокращением `<fqdn>` (fully qualified domain name, полное доменное имя). Встретив его, замените его, в зависимости от обстоятельств, на адрес интересующего вас сайта или сервера, например, на нечто вроде `server-name.company.com`.

Ping

Утилита `ping`, как можно судить по её названию, используется для проверки связи между узлами сети, между компьютером, на котором её запускают, и другой системой. Эта утилита использует протокол ICMP, отправляя эхо-запросы, на которые отвечает удалённая система, получающая их.

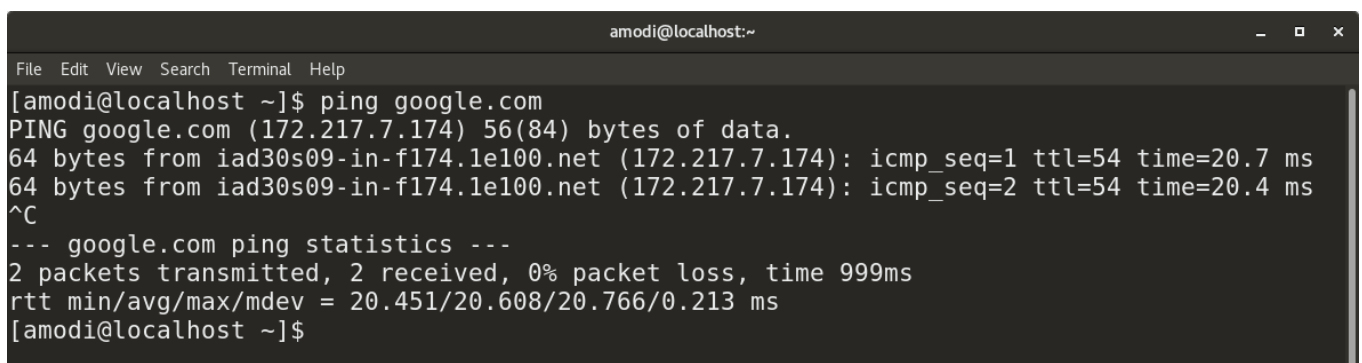
Использование `ping`, кроме того — это хороший способ проверки связности сети, проводимой в качестве первого шага диагностики сети при наличии неполадок. Команду `ping` можно использовать с адресами IPv4 и IPv6. Тут можно почитать подробности об IP-адресах и о работе с ними.

■ Примеры

IPv4: `ping <ip address>/<fqdn>`

IPv6: `ping6 <ip address>/<fqdn>`

Ping, кроме того, можно использовать для выяснения IP-адресов сайтов на основе их имён. Вот как это выглядит.



```
amodi@localhost:~  
File Edit View Search Terminal Help  
[amodi@localhost ~]$ ping google.com  
PING google.com (172.217.7.174) 56(84) bytes of data.  
64 bytes from iad30s09-in-f174.1e100.net (172.217.7.174): icmp_seq=1 ttl=54 time=20.7 ms  
64 bytes from iad30s09-in-f174.1e100.net (172.217.7.174): icmp_seq=2 ttl=54 time=20.4 ms  
^C  
--- google.com ping statistics ---  
2 packets transmitted, 2 received, 0% packet loss, time 999ms  
rtt min/avg/max/mdev = 20.451/20.608/20.766/0.213 ms  
[amodi@localhost ~]$
```

Использование `ping` для выяснения IP-адреса сайта по его имени

Traceroute

Traceroute — это приятная утилита, которая позволяет исследовать маршруты передачи данных между компьютерами. В то время как команда `ping` направлена на то, чтобы выяснить, можно ли установить связь между двумя узлами сети, `traceroute` даёт сведения об IP-адресах маршрутизаторов, через которые проходят данные от вашей системы до конечной, например — до веб-сайта или сервера. Команда `traceroute` обычно применяется на втором шаге диагностики сети, после команды `ping`.

■ Пример

```
traceroute <ip address>/<fqdn>
```

Telnet

Утилита `telnet` позволяет связаться с удалённым компьютером по протоколу Telnet и взаимодействовать с ним, используя соответствующие команды.

■ Пример

Для организации сеанса Telnet-связи с другим компьютером используется следующая команда:

```
telnet <ip address>/<fqdn>
```

Netstat

Эта команда позволяет собирать сведения о сети и используется в ходе поиска и исправления сетевых неполадок, применяется для проверки данных о работе интерфейсов и портов, для исследования таблиц маршрутизации, для изучения информации о работе протоколов. Эта команда непременно должна присутствовать в арсенале системного администратора.

■ Примеры

Для того чтобы получить список всех портов, находящихся в режиме прослушивания, воспользуйтесь такой командой:

```
netstat -l
```

Следующая команда выводит сведения обо всех портах. Для того чтобы ограничиться только TCP-портами, нужно воспользоваться ключом `-at`, для того, чтобы получить данные об UDP-портах, используйте ключ `-au`.

```
netstat -a
```

Для просмотра таблиц маршрутизации воспользуйтесь такой командой:

```
netstat -r
```

Вот как выглядит результат выполнения этой команды.

```
stack@undercloud-0:~  
File Edit View Search Terminal Help  
[stack@undercloud-0 ~]$ netstat -r  
Kernel IP routing table  
Destination      Gateway          Genmask         Flags   MSS Window  irtt  Iface  
default          gateway          0.0.0.0         UG      0 0       0     eth1  
10.0.0.0          0.0.0.0          255.255.255.0   U       0 0       0     eth2  
172.16.0.0        0.0.0.0          255.255.255.0   U       0 0       0     eth1  
172.17.0.0        0.0.0.0          255.255.0.0     U       0 0       0     docker0  
192.168.24.0      0.0.0.0          255.255.255.0   U       0 0       0     br-ctlplane  
[stack@undercloud-0 ~]$
```

Сведения о таблице маршрутизации

Вот вариант этой команды, выводящий статистику по протоколам:

```
netstat -s
```

```
stack@undercloud-0:~  
File Edit View Search Terminal Help  
[stack@undercloud-0 ~]$ netstat -s  
Ip:  
  51869585 total packets received  
  230846 forwarded  
  0 incoming packets discarded  
  51638713 incoming packets delivered  
  51483892 requests sent out  
  16 outgoing packets dropped  
Icmp:  
  3374 ICMP messages received  
  0 input ICMP message failed.  
  ICMP input histogram:  
    destination unreachable: 3356  
    echo requests: 7  
    echo replies: 1  
    timestamp request: 4
```

Статистика по протоколам

Следующий вариант вызова `netstat` позволяет узнать сведения об отправленных и полученных пакетах (transmission/receive, TX/RX) по каждому интерфейсу:

```
stack@undercloud-0:~$ netstat -i
Kernel Interface table
Iface      MTU      RX-OK RX-ERR RX-DRP RX-OVR      TX-OK TX-ERR TX-DRP TX-OVR Flg
br-ctlpl   1500    1279020      0      0 0      1328553      0      0      0 BMRU
docker0    1500      0      0      0 0          0      0      0      0 BMU
eth0       1500    3901543      0     14 0     3969814      0      0      0 BMRU
eth1       1500    2713560      0      2 0     1675657      0      0      0 BMRU
eth2       1500    305751      0      2 0       27421      0      0      0 BMRU
lo         65536 45856863      0      0 0    45856863      0      0      0 LRU
[stack@undercloud-0 ~]$
```

Данные об отправленных и полученных пакетах

Nmcli

Утилита `nmcli` отлично подходит для управления сетевыми соединениями, для выполнения настроек и для решения других подобных задач. С её помощью можно управлять программой `NetworkManager` и модифицировать сетевые параметры различных устройств.

■ Примеры

Вот как с помощью `nmcli` вывести список сетевых интерфейсов:

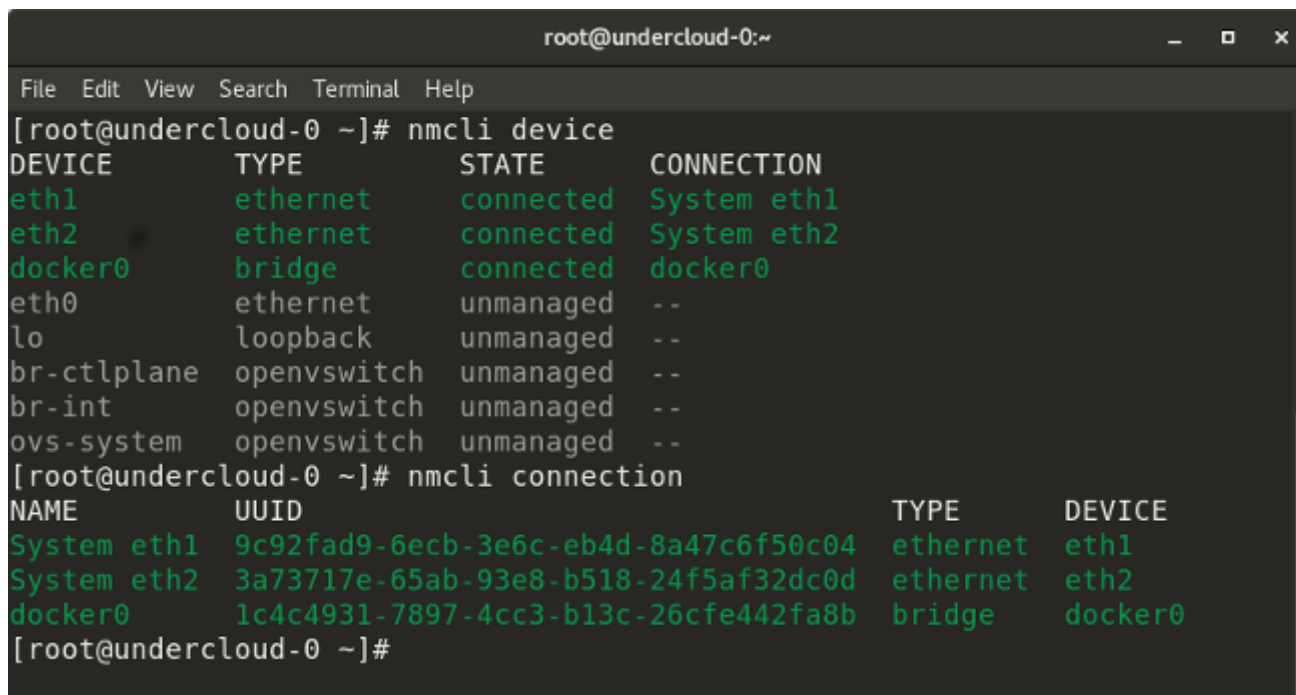
```
nmcli device
```

Так можно вывести информацию по конкретному интерфейсу:

```
nmcli device show <interface>
```

Следующий вариант вызова команды позволяет проверить подключение устройства к сети:

```
nmcli connection
```



```
root@undercloud-0:~  
File Edit View Search Terminal Help  
[root@undercloud-0 ~]# nmcli device  
DEVICE          TYPE          STATE          CONNECTION  
eth1             ethernet      connected      System eth1  
eth2             ethernet      connected      System eth2  
docker0          bridge        connected      docker0  
eth0             ethernet      unmanaged      --  
lo               loopback      unmanaged      --  
br-ctlplane      openvswitch   unmanaged      --  
br-int           openvswitch   unmanaged      --  
ovs-system       openvswitch   unmanaged      --  
[root@undercloud-0 ~]# nmcli connection  
NAME            UUID                                TYPE          DEVICE  
System eth1     9c92fad9-6ecb-3e6c-eb4d-8a47c6f50c04 ethernet      eth1  
System eth2     3a73717e-65ab-93e8-b518-24f5af32dc0d ethernet      eth2  
docker0         1c4c4931-7897-4cc3-b13c-26cfe442fa8b bridge        docker0  
[root@undercloud-0 ~]#
```

Примеры использования nmcli

Эта команда позволяет отключить заданный интерфейс:

```
nmcli connection down <interface>
```

А эта позволяет включить интерфейс:

```
nmcli connection up <interface>
```

Вот пример команды, которая добавляет VLAN-интерфейс с заданным VLAN-номером, IP-адресом и шлюзом к указанному интерфейсу:

```
nmcli con add type vlan con-name <connection-name> dev <interface> id  
<vlan-number> ipv4 <ip/cidr> gw4 <gateway-ip>
```

Маршрутизация

Существует множество команд, которые можно использовать для проверки правил маршрутизации и их настройки. Рассмотрим самые полезные из них.

■ Примеры

Следующая команда показывает все текущие маршруты, настроенные для соответствующих интерфейсов:

```
ip route
```



```
stack@undercloud-0:~  
File Edit View Search Terminal Help  
[stack@undercloud-0 ~]$ ip route  
default via 172.16.0.1 dev eth1 proto dhcp metric 101  
10.0.0.0/24 dev eth2 proto kernel scope link src 10.0.0.37 metric 102  
172.16.0.0/24 dev eth1 proto kernel scope link src 172.16.0.4 metric 101  
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1  
192.168.24.0/24 dev br-ctlplane proto kernel scope link src 192.168.24.1  
[stack@undercloud-0 ~]$  
[stack@undercloud-0 ~]$  
[stack@undercloud-0 ~]$
```

Маршруты, настроенные для интерфейсов

Эта команда позволяет добавить в таблицу маршрутизации шлюз, используемый по умолчанию:

```
route add default gw <gateway-ip>
```

Следующая команда добавляет в таблицу маршрутизации новый сетевой маршрут. Существует и множество других её параметров, позволяющих выполнять такие операции, как добавление маршрута и шлюза, используемых по умолчанию, и так далее.

```
route add -net <network ip/cidr> gw <gateway ip> <interface>
```

С помощью такой команды можно удалить запись о заданном маршруте из таблицы маршрутизации:

```
route del -net <network ip/cidr>
```

Вот примеры использования команды `route`.

```

stack@undercloud-0:~$ sudo route add -net 10.0.2.0/24 gw 10.0.0.1 eth2
stack@undercloud-0:~$ route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0          172.16.0.1     0.0.0.0         UG      101    0      0 eth1
10.0.0.0          0.0.0.0        255.255.255.0   U       102    0      0 eth2
10.0.2.0          10.0.0.1       255.255.255.0   UG      0      0      0 eth2
172.16.0.0        0.0.0.0        255.255.255.0   U       101    0      0 eth1
172.17.0.0        0.0.0.0        255.255.0.0     U       0      0      0 docker0
192.168.24.0      0.0.0.0        255.255.255.0   U       0      0      0 br-ctlplane
stack@undercloud-0:~$ sudo route del -net 10.0.2.0/24
stack@undercloud-0:~$ route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0          172.16.0.1     0.0.0.0         UG      101    0      0 eth1
10.0.0.0          0.0.0.0        255.255.255.0   U       102    0      0 eth2
172.16.0.0        0.0.0.0        255.255.255.0   U       101    0      0 eth1
172.17.0.0        0.0.0.0        255.255.0.0     U       0      0      0 docker0
192.168.24.0      0.0.0.0        255.255.255.0   U       0      0      0 br-ctlplane
stack@undercloud-0:~$

```

Использование команды *route*

Вот команда, которая применяется для вывода текущей таблицы соседей.

Кроме того, её можно использовать для добавления, изменения или удаления сведений о соседях:

`ip neighbor`

Взглянем на примеры её использования.

```

stack@undercloud-0:~$ ip neighbor
10.0.0.215 dev eth2 lladdr fa:16:3e:91:da:4a STALE
10.0.0.220 dev eth2 lladdr fa:16:3e:91:da:4a STALE
10.0.0.107 dev eth2 lladdr 52:54:00:d1:e0:8a STALE
172.16.0.1 dev eth1 lladdr 52:54:00:70:43:c9 REACHABLE
192.168.24.10 dev br-ctlplane lladdr 52:54:00:41:20:7a REACHABLE
10.0.0.1 dev eth2 lladdr 52:54:00:a9:04:7b STALE
10.0.0.103 dev eth2 lladdr 52:54:00:d1:e0:8a STALE
192.168.24.13 dev br-ctlplane lladdr 52:54:00:e2:12:8a REACHABLE
192.168.24.8 dev br-ctlplane lladdr 52:54:00:11:e7:2b REACHABLE
fe80::5054:ff:fea9:47b dev eth2 lladdr 52:54:00:a9:04:7b router STALE
2620:52:0:13b8::fe dev eth2 lladdr 52:54:00:a9:04:7b router STALE
stack@undercloud-0:~$

```

Данные, полученные с помощью команды *ip neighbor*

Вот сведения о команде `ip neigh`

```
stack@undercloud-0:~  
File Edit View Search Terminal Help  
[stack@undercloud-0 ~]$ ip neigh help  
Usage: ip neigh { add | del | change | replace }  
        { ADDR [ lladdr LLADDR ] [ nud STATE ] | proxy ADDR } [ dev DEV ]  
        ip neigh { show | flush } [ proxy ] [ to PREFIX ] [ dev DEV ] [ nud STATE ]  
        [ vrf NAME ]  
  
STATE := { permanent | noarp | stale | reachable | none |  
          incomplete | delay | probe | failed }  
[stack@undercloud-0 ~]$  
[stack@undercloud-0 ~]$
```

Сведения о команде `ip neigh`

Команда `arp` (ARP — это сокращение от Address Resolution Protocol, протокол определения адреса) похожа на `ip neighbor`. Утилита `arp` выводит данные о соответствии IP-адресов MAC -адресам. Вот как её использовать:

`arp`

Вот пример её вызова.

```
stack@undercloud-0:~  
File Edit View Search Terminal Help  
[stack@undercloud-0 ~]$ arp  
Address          HWtype  HWaddress      Flags Mask    Iface  
10.0.0.215       ether   fa:16:3e:91:da:4a  C             eth2  
10.0.0.220       ether   fa:16:3e:91:da:4a  C             eth2  
10.0.0.107       ether   52:54:00:d1:e0:8a  C             eth2  
gateway          ether   52:54:00:70:43:c9  C             eth1  
192.168.24.10    ether   52:54:00:41:20:7a  C             br-ctlplane
```

Вызов команды `arp`

Tcpdump и Wireshark

Linux даёт в распоряжение администратора множество инструментов для захвата и анализа пакетов. Среди них, например, `tcpdump`, `wireshark`, `tshark`, и другие. Они используются для захвата сетевого трафика в передаваемых системой пакетах или в пакетах, получаемых ей. Это делает их весьма ценным инструментом администратора, помогающим в деле выяснения причин различных сетевых неполадок. Тем, кто предпочитает командную строку всем остальным способам общения с компьютерами, понравится `tcpdump`. Тем же, кто любит графические интерфейсы, можно порекомендовать `wireshark` — отличный инструмент для захвата и анализа пакетов. Утилита `tcpdump` — это встроенное в Linux средство для захвата сетевого трафика. Его можно использовать для захвата и вывода трафика с фильтрацией по портам, протоколам, и по другим признакам.

■ Примеры

Такая команда показывает, в режиме реального времени, пакеты с заданного интерфейса:

```
tcpdump -i <interface-name>
```

Пакеты можно сохранять в файл, воспользовавшись флагом `-w` и задав имя файла:

```
tcpdump -w <output-file.> -i <interface-name>
```

Вот пример использования `tcpdump`.

```
stack@undercloud-0 ~$ sudo tcpdump -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
01:07:26.033020 STP 802.1d, Config, Flags [none], bridge-id 8000.52:54:00:5b:86:3e.8004, length 35
01:07:28.033025 STP 802.1d, Config, Flags [none], bridge-id 8000.52:54:00:5b:86:3e.8004, length 35
01:07:30.033080 STP 802.1d, Config, Flags [none], bridge-id 8000.52:54:00:5b:86:3e.8004, length 35
01:07:32.033072 STP 802.1d, Config, Flags [none], bridge-id 8000.52:54:00:5b:86:3e.8004, length 35
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
[stack@undercloud-0 ~]$
```

Использование *tcpdump*

Следующий вариант команды используется для захвата пакетов, приходящих с заданного IP системы-источника:

```
tcpdump -i <interface> src <source-ip>
```

Так можно захватить пакеты, идущие на заданный адрес системы-приёмника:

```
tcpdump -i <interface> dst <destination-ip>
```

Вот пример использования *tcpdump* для захвата пакетов для заданного номера порта, например, это может быть порт 53, 80, 8080, и так далее:

```
tcpdump -i <interface> port <port-number>
```

Здесь показано, как с помощью *tcpdump* захватывать пакеты заданного протокола, вроде TCP, UDP или других:

```
tcpdump -i <interface> <protocol>
```

Iptables

Утилита `iptables` похожа на файрвол, она поддерживает фильтрацию пакетов, что позволяет управлять трафиком, пропуская или блокируя его. Диапазон возможностей этой утилиты огромен. Рассмотрим несколько наиболее распространённых вариантов её использования.

■ Примеры

Следующая команда позволяет вывести все существующие правила `iptables`:

```
iptables -L
```

Эта команда удаляет все существующие правила:

```
iptables -F
```

Следующие команды разрешают прохождение трафика с заданного номера порта к заданному интерфейсу:

```
iptables -A INPUT -i <interface> -p tcp -dport <port-number> -m state  
-state NEW,ESTABLISHED -j ACCEPT
```

```
iptables -A OUTPUT -o <interface> -p tcp -sport <port-number> -m state  
- state ESTABLISHED -j ACCEPT
```

Следующие команды разрешают loopback-доступ к системе:

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

Nslookup

Инструмент `nslookup` используется для получения сведений о назначении IP-адресов сетевым ресурсам. Его можно использовать и для получения сведений с DNS-серверов, например таких, как все DNS-записи для некоего веб-сайта (ниже мы рассмотрим соответствующий пример). На `nslookup` похожа утилита `dig` (Domain Information Groper).

■ Примеры

Следующая команда выводит IP-адреса вашего DNS-сервера в поле Server, и, ниже, выдаёт IP-адрес искомого сайта:

```
nslookup <website-name.com>
```

Такая команда показывает все доступные записи для заданного веб-сайта или домена:

```
nslookup -type=any <website-name.com>
```

Поиск неполадок

Вот набор команд и список важных файлов, используемых для идентификации сетевых неполадок.

■ Примеры

- `ss` — утилита для вывода статистической информации о сокетах.
- `nmap <ip-address>` — имя этой команды является сокращением от Network Mapper. Она сканирует сетевые порты, обнаруживает хосты, выясняет MAC-адреса и выполняет множество других задач.
- `ip addr/ifconfig -a` — эта команда предоставляет сведения об IP-адресах и другие данные по всем интерфейсам системы.
- `ssh -vvv user@<ip/domain>` — такая команда позволяет подключиться по SSH к другому компьютеру, используя заданный IP-адрес или доменное имя компьютера и имя пользователя. Флаг `-vvv` позволяет получать подробные сведения о происходящем.
- `ethtool -S <interface>` — данная команда позволяет вывести статистические сведения по заданному интерфейсу.
- `ifup <interface>` — эта команда включает указанный интерфейс.
- `ifdown <interface>` — эта команда отключает указанный интерфейс.
- `systemctl restart network` — с помощью этой команды можно перезагрузить системную сетевую подсистему.
- `/etc/sysconfig/network-scripts/<interface-name>` — это — файл настройки интерфейсов, используемый для указания IP-адреса, сети,

шлюза и других параметров для заданного интерфейса. Здесь можно задать использование интерфейсом DHCP-режима.

- `/etc/hosts` — данный файл содержит сведения о соответствии хостов или доменов IP-адресам, настроенные администратором.
- `/etc/resolv.conf` — в этом файле хранятся настройки DNS.
- `/etc/ntp.conf` — этот файл хранит настройки NTP.