

ASTRA LINUX GUIDE

TCPDUMP, HEXDUMP, NETSTAT, SS, TELNET, NMAP

Команда Tcpdump в Linux

tcpdump — утилита командной строки, которую вы можете использовать для захвата и проверки сетевого трафика, поступающего в вашу систему и из нее. Это наиболее часто используемый среди сетевых администраторов инструмент для устранения неполадок в сети и тестирования безопасности.

Несмотря на его название tcpdump, вы также можете захватывать не только TCP-трафик, а и UDP, ARP или ICMP. Захваченные пакеты могут быть записаны в файл или стандартный вывод. Одной из самых мощных функций команды tcpdump является ее способность использовать фильтры и собирать только те данные, которые вы хотите проанализировать.

Установка tcpdump

tcpdump устанавливается по умолчанию в большинстве дистрибутивов Linux и macOS. Чтобы проверить, доступна ли команда tcpdump в вашей системе, введите:

```
tcpdump --version
```

Вывод должен выглядеть примерно так:

```
tcpdump version 4.9.2
libpcap version 1.8.1
OpenSSL 1.1.1b  26 Feb 2019
```

Если tcpdump нет в вашей системе, вышеприведенная команда выведет “tcpdump: command not found”. Вы можете легко установить tcpdump, используя менеджер пакетов вашего дистрибутива.

Установка tcpdump на Ubuntu и Debian

```
sudo apt update && sudo apt install tcpdump
```

Захват пакетов с tcpdump

Общий синтаксис команды tcpdump выглядит следующим образом:

```
tcpdump [options] [expression]
```

Команда options позволяет вам контролировать поведение команды.

Фильтр expression определяет, какие пакеты будут захвачены.

Только root или пользователь с sudo привилегиями может работать tcpdump. Если вы попытаетесь запустить команду как непrivилегированный пользователь, вы получите сообщение об ошибке: “You don't have permission to capture on that device”.

ASTRA LINUX GUIDE

tcpdump

Самый простой вариант использования — вызвать tcpdump без каких-либо опций и фильтров:

```
sudo tcpdump
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens3, link-type EN10MB (Ethernet), capture size 262144 bytes
15:47:24.248737 IP linuxize-host.ssh > desktop-machine.39196: Flags [P.], seq
201747193:201747301, ack 1226568763, win 402, options [nop,nop,TS val 1051794587 ecr
2679218230], length 108
15:47:24.248785 IP linuxize-host.ssh > desktop-machine.39196: Flags [P.], seq 108:144, ack 1, win
402, options [nop,nop,TS val 1051794587 ecr 2679218230], length 36
15:47:24.248828 IP linuxize-host.ssh > desktop-machine.39196: Flags [P.], seq 144:252, ack 1, win
402, options [nop,nop,TS val 1051794587 ecr 2679218230], length 108
```

... Long output suppressed

```
23116 packets captured
23300 packets received by filter
184 packets dropped by kernel
```

tcpdump будет продолжать захват пакетов и запись в стандартный вывод, пока не получит сигнал прерывания. Используйте комбинацию клавиш Ctrl+C, чтобы отправить сигнал прерывания и остановить команду.

Для более подробного вывода передайте параметр **-v** или **-vv** для еще более подробного вывода:

```
sudo tcpdump -vv
```

Вы можете указать количество пакетов для перехвата, используя опцию **-c**. Например, чтобы захватить только десять пакетов, вы должны набрать:

```
sudo tcpdump -c 10
```

После захвата пакетов tcpdump остановится.

Когда интерфейс не указан, tcpdump используется первый найденный интерфейс и дамп всех пакетов, проходящих через этот интерфейс.

Используйте опцию **-D**, чтобы напечатать список всех доступных сетевых интерфейсов, с которых tcpdump может собирать пакеты:

```
sudo tcpdump -D
```

Для каждого интерфейса команда выводит имя интерфейса, краткое описание и связанный индекс (номер):

```
1.eth0 [Up, Running]  
2.any (Pseudo-device that captures on all interfaces) [Up, Running]  
3.lo [Up, Running, Loopback]
```

Приведенные выше результаты показывают, что `eth0` это первый интерфейс, найденный `tcpdump` используемый, когда для команды не предоставлен интерфейс. Второй интерфейс `any`- это специальное устройство, которое позволяет захватывать все активные интерфейсы.

Чтобы указать интерфейс, на котором вы хотите захватывать трафик, вызовите команду с параметром `-i`, за которым следует имя интерфейса или связанный индекс. Например, чтобы захватить все пакеты со всех интерфейсов, вы должны указать интерфейс `any`:

```
sudo tcpdump -i any
```

По умолчанию `tcpdump` выполняет обратное разрешение DNS для IP-адресов и переводит номера портов в имена. Используйте опцию `-n`, чтобы отключить перевод:

```
sudo tcpdump -n
```

Пропуск поиска DNS позволяет избежать генерации трафика DNS и делает вывод более читабельным. Рекомендуется использовать эту опцию каждый раз, когда вы вызываете `tcpdump`.

Вместо отображения вывода на экране вы можете перенаправить его в файл, используя операторы перенаправления `>` и `>>`:

```
sudo tcpdump -n -i any > file.out
```

Вы также можете просмотреть данные при сохранении в файл с помощью команды `tee`:

```
sudo tcpdump -n -l | tee file.out
```

Опция `-l` в команде выше говорит `tcpdump` чтобы сделать вывод строки в буфер. Когда эта опция не используется, вывод не будет записан на экране при создании новой строки.

Понимание вывода tcpdump

tcpdump выводит информацию для каждого захваченного пакета на новой строке.

Каждая строка содержит метку времени и информацию об этом пакете, в зависимости от протокола.

Типичный формат строки протокола TCP выглядит следующим образом:

[Timestamp] [Protocol] [Src IP].[Src Port] > [Dst IP].[Dst Port]: [Flags], [Seq], [Ack], [Win Size], [Options], [Data Length]

Давайте посмотрим поле за полем и объясним следующую строку:

15:47:24.248737 IP 192.168.1.185.22 > 192.168.1.150.37445: Flags [P.], seq 201747193:201747301, ack 1226568763, win 402, options [nop,nop,TS val 1051794587 ecr 2679218230], length 108

- 15:47:24.248737- Отметка времени захваченного пакета указывается по местному времени и использует следующий формат: hours:minutes:seconds.frac где frac — доли секунды с полуночи.
- IP- Пакетный протокол. В этом случае IP означает интернет-протокол версии 4 (IPv4).
- 192.168.1.185.22- IP-адрес источника и порт, разделенные точкой (.).
- 192.168.1.150.37445- IP-адрес и порт назначения, разделенные точкой (.).
- Flags [P.]- Поле Флаги TCP. В этом примере [P.] означает пакет Push Acknowledgement, который используется для подтверждения предыдущего пакета и отправки данных. Другие типичные значения полей флага:
 - [.] — ACK (Подтверждение)
 - [S] — SYN (Начать соединение)
 - [P] — PSH (Push-данные)
 - [F] — FIN (Завершить соединение)
 - [R] — RST (Сброс соединения)
 - [S.] — SYN-ACK (Пакет SynAcK)
- seq 201747193:201747301- Порядковый номер в обозначении first:last. Показывает количество данных, содержащихся в пакете. За исключением первого пакета в потоке данных, где эти числа являются абсолютными, все последующие пакеты используют в качестве относительных позиций байтов. В этом примере это число 201747193:201747301 означает, что этот пакет содержит байты с 201747193 по 201747301 потока данных. Используйте опцию -S для печати абсолютных порядковых номеров.
- ack 1226568763 Номер подтверждения — это порядковый номер следующих данных, ожидаемых другим концом этого соединения.
- win 402 — Номер окна — это количество доступных байтов в приемном буфере.
- options [nop,nop,TS val 1051794587 ecr 2679218230]- Варианты TCP. пор или «без операции» — это заполнение, используемое, чтобы сделать заголовок TCP кратным 4 байтам. TS val является отметкой времени TCP и естественно означает эхо-ответ. Посетите документацию IANA для получения дополнительной информации о параметрах TCP.
- length 108 — длина данных полезной нагрузки

Фильтры tcpdump

Когда tcpdump вызывается без фильтров, он захватывает весь трафик и выдает огромное количество выходных данных, что очень затрудняет поиск и анализ пакетов, представляющих интерес.

Фильтры являются одной из самых мощных функций команды tcpdump. Они позволяют вам захватывать только те пакеты, которые соответствуют выражению. Например, при устранении неполадок, связанных с веб-сервером, вы можете использовать фильтры для получения только HTTP-трафика.

tcpdump использует синтаксис Berkeley Packet Filter (BPF) для фильтрации захваченных пакетов с использованием различных параметров обработки, таких как протоколы, IP-адреса источника и назначения и порты и т. д.

В этой статье мы рассмотрим некоторые из наиболее распространенных фильтров. Для получения списка всех доступных фильтров проверьте страницу руководства pcap-filter.

Фильтрация по протоколу

Чтобы ограничить перехват определенным протоколом, укажите протокол в качестве фильтра. Например, для захвата только UDP-трафика вы будете использовать:

```
sudo tcpdump -n udp
```

Другой способ определить протокол — использовать спецификатор proto, за которым следует номер протокола. Следующая команда отфильтрует протокол № 17 и выдаст тот же результат, что и приведенный выше:

```
sudo tcpdump -n proto 17
```

Для получения дополнительной информации о номерах, проверьте список номеров протокола IP.

Фильтрация по хосту

Для захвата только пакетов, связанных с конкретным хостом, используйте спецификатор host:

```
sudo tcpdump -n host 192.168.1.185
```

Хост может быть либо IP-адресом, либо именем.

Вы также можете отфильтровать выходные данные по заданному диапазону IP-адресов с помощью квалификатора net. Например, для выгрузки только пакетов, связанных с 10.10.0.0/16, используйте:

```
sudo tcpdump -n net 10.10
```

Фильтрация по порту

Чтобы ограничить захват только пакетами от определенного порта, используйте классификатор port. Команда ниже захватывает пакеты, связанные со службой SSH (порт 22), с помощью этой команды:

```
sudo tcpdump -n port 22
```

Классификатор portrange позволяет захватывать трафик в диапазоне портов:

```
sudo tcpdump -n portrange 110-150
```

Фильтрация по источнику и назначению

Вы можете также фильтровать пакеты на основе источника или порта назначения или хост, используя src, dst, src, dst и src или классификатор dst.

Следующая команда перехватывает приходящие пакеты с хоста с IP 192.168.1.185:

```
sudo tcpdump -n src host 192.168.1.185
```

Чтобы найти трафик, поступающий из любого источника на порт 80, вы должны использовать:

```
sudo tcpdump -n dst port 80
```

Комплексные фильтры

Фильтры можно комбинировать с помощью операторов and(&&), or(||) и not(!).

Например, чтобы перехватить весь HTTP-трафик с IP-адреса источника 192.168.1.185, вы должны использовать эту команду:

```
sudo tcpdump -n src 192.168.1.185 and tcp port 80
```

Вы также можете использовать скобки для группировки и создания более сложных фильтров:

```
sudo tcpdump -n 'host 192.168.1.185 and (tcp port 80 or tcp port 443)'
```

Чтобы избежать ошибок синтаксического анализа при использовании специальных символов, заключите фильтры в одинарные кавычки.

Вот еще один пример команды для захвата всего трафика, кроме SSH, с исходного IP-адреса 192.168.1.185:

```
sudo tcpdump -n src 192.168.1.185 and not dst port 22
```

Инспекция пакетов

По умолчанию tcpdump захватывает только заголовки пакетов. Однако иногда вам может понадобиться проверить содержимое пакетов.

tcpdump позволяет печатать содержимое пакетов в ASCII и HEX.

Опция -A указывает tcpdump на печать каждого пакета в ASCII и -x в HEX:

```
sudo tcpdump -n -A
```

Чтобы показать содержимое пакета в HEX и ASCII, используйте параметр -X:

```
sudo tcpdump -n -X
```

Чтение и запись снимков в файл

Еще одна полезная функция tcpdump, это записывать пакеты в файл. Это удобно, когда вы захватываете большое количество пакетов или когда хотите захватить пакеты для последующего анализа.

Чтобы начать запись в файл, используйте параметр -w, за которым следует выходной файл захвата:

```
sudo tcpdump -n -w data.pcap
```

Эта команда выше сохранит захват в файл с именем data.pcap. Вы можете назвать файл так, как хотите, но это распространенное соглашение — использовать расширение .pcap (захват пакета).

Когда -w используется опция, вывод не отображается на экране. tcpdump записывает необработанные пакеты и создает двоичный файл, который невозможно прочитать с помощью обычного текстового редактора.

Чтобы проверить содержимое файла, вызовите tcpdump с опцией -r:

```
sudo tcpdump -r data.pcap
```

Файл захвата также можно проверить с помощью других инструментов анализа пакетов, таких как Wireshark.

При захвате пакетов в течение длительного периода времени вы можете включить ротацию файлов. tcpdump позволяет создавать новые файлы или вращать файл дампа за указанный промежуток времени или фиксированный размер. Следующая команда будет создавать до десяти 200МБ файлов, именованных file.pcap0, file.pcap1 и так далее: перед перезаписью старых файлов.

```
sudo tcpdump -n -W 10 -C 200 -w /tmp/file.pcap
```

После создания десяти файлов старые файлы будут перезаписаны.

Обратите внимание, что вы должны работать tcpdump только во время устранения неполадок.

Если вы хотите начать tcpdump в определенное время, вы можете использовать cronjob . tcpdump не имеет возможности выйти через заданное время. Вы можете использовать timeout команду, чтобы остановить tcpdump через некоторое время. Например, чтобы выйти через 5 минут, вы должны использовать:

```
sudo timeout 300 tcpdump -n -w data.pcap &
```

Символ амперсанда (&) в конце команды запускает команду в фоновом режиме.

Вывод

tcpdump является инструментом командной строки для анализа и устранения проблем, связанных с сетью.

HEXDUMP

Hexdump — мощный инструмент в системах Linux, который в основном используется разработчиками и отладчиками приложений. Он может преобразовывать входные файлы и данные в приятный и читаемый формат.

Вот пример из реальной жизни, где может быть полезен hexdump. Если вы работаете с двоичными данными, это будет очень сложно понять. Для удобства вы можете быстро преобразовать двоичные данные в шестнадцатеричные или десятичные.

Hexdump в Linux

Hexdump — это простая, но полезная программа, написанная на языке С. Вот почему профессиональные программисты на С могут легко использовать его. Однако, даже если у вас нет опыта программирования на С, вы все равно можете использовать hexdump для своих целей.

Hexdump предустановлен в любом дистрибутиве Linux. В этой статье мы будем использовать Ubuntu в демонстрационных целях.

Использование Hexdump

Для демонстрации использования hexdump я создал образец текстового файла dummy.txt.

```
$ cat dummy.txt
```

Мы передадим этот файл в hexdump, чтобы преобразовать его содержимое в различные форматы вывода.

Однобайтовый восьмеричный

Следующая команда hexdump распечатает входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 16 байтов входных данных, разделенных пробелами, каждый из которых имеет 3 столбца и заполнен нулями в восьмеричном формате.

```
$ hexdump -b <input_file_content>
```

Отображение однобайтовых символов

Следующая команда hexdump отобразит входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 16 символов входных данных, разделенных пробелами, каждый из которых имеет 3 столбца и заполнен пробелами.

```
$ hexdump -c <input_file_content>
```

Канонический шестнадцатеричный + ASCII

Следующая команда hexdump отобразит входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 16 шестнадцатеричных байтов, разделенных пробелами, по 2 столбца в каждом. Следующее содержимое будет теми же байтами в формате %_p , заключенными в «|» символы.

```
$ hexdump -C <input_file_content>
```

ASTRA LINUX GUIDE

hexdump

Двухбайтовый десятичный

Следующая команда hexdump отобразит входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 8 разделенных пробелом двухбайтовых блоков входных данных, каждая из которых имеет 5 столбцов и заполнена нулями в десятичном формате без знака.

```
$ hexdump -d <input_file_content>
```

Двухбайтовый восьмеричный

Следующая команда hexdump распечатает входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 8 разделенных пробелами 2 байта входных данных, каждый из которых содержит 6 столбцов и заполнен нулями в восьмеричном формате.

```
$ hexdump -o <input_file_content>
```

Двухбайтовый шестнадцатеричный

Следующая команда hexdump распечатает входные данные в шестнадцатеричном формате. В выводе каждая строка содержит 8 разделенных пробелами 2 байта входных данных, каждый с 4 столбцами и заполненными нулями в шестнадцатеричном формате.

```
$ hexdump -x <input_file_content>
```

Показать весь ввод

При использовании шестнадцатеричного дампа он заменяет содержимое повторяющихся строк одной звездочкой. Если вы хотите заставить hexdump выводить все содержимое, используйте флаг «-v».

```
$ cat dummy.txt
```

```
$ hexdump -b dummy.txt
```

```
$ hexdump -v -b <input_file_content>
```

Ограничить количество байтов

Hexdump поддерживает опцию определения определенного количества байтов из файла в hexdump. Чтобы указать количество, используйте флаг «-s», за которым следует количество байтов.

```
$ hexdump -s 2 -c <input_file_content>
```

ASTRA LINUX GUIDE

netstat

NETSTAT

Команда Netstat отображает различные сетевые данные, такие как сетевые подключения, таблица маршрутизации, статистики интерфейсов, маскированные соединения, многоадресное пространство и т.д.,

1. Список всех портов (как прослушиваемые, так и не прослушиваемые порты)

Список всех портов с помощью команды Netstat -a

```
# netstat -a | more
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	localhost:30037	*.*	LISTEN
udp	0	0	*:bootpc	*.*	

Active UNIX domain sockets (servers and established)

Proto	RefCount	Flags	Type	State	I-Node	Path
unix	2	[ACC]	STREAM	LISTENING	6135	/tmp/X11-unix/X0
unix	2	[ACC]	STREAM	LISTENING	5140	/var/run/acpid.socket

Список всех портов TCP с помощью netstat -at

```
# netstat -at
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:http	0.0.0.0:*	LISTEN
tcp	0	0	localhost:webcache	0.0.0.0:*	LISTEN
tcp	0	0	andreyex.ru:domain	0.0.0.0:*	LISTEN
tcp	0	0	localhost:domain	0.0.0.0:*	LISTEN

Перечисление всех UDP-портов с помощью netstat -au

```
# netstat -au
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
udp	0	0	andreyex.ru:49419	google-public-dn:domain	ESTABLISHED
udp	0	0	andreyex.ru:39293	google-public-dn:domain	ESTABLISHED
udp	0	0	andreyex.ru:50053	google-public-dn:domain	ESTABLISHED

2. Список сокетов, которые находятся в состояние прослушивания

Список только прослушивающих портов с помощью netstat -l

```
# netstat -l
```

Active Internet connections (only servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:http	0.0.0.0:*	LISTEN
tcp	0	0	localhost:webcache	0.0.0.0:*	LISTEN
tcp	0	0	andreyex.ru:domain	0.0.0.0:*	LISTEN
tcp	0	0	localhost:domain	0.0.0.0:*	LISTEN

ASTRA LINUX GUIDE

netstat

Список только прослушивающихся TCP портов с помощью netstat -lt

```
# netstat -lt
```

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 0.0.0.0:http           0.0.0.0:*
tcp      0      0 localhost:webcache     0.0.0.0:*
tcp      0      0 andreyex.ru:domain    0.0.0.0:*
tcp      0      0 localhost:domain      0.0.0.0:*
```

Список только прослушивающихся UDP портов с помощью netstat -lu

```
# netstat -lu
```

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
udp      0      0 andreyex.ru:domain    0.0.0.0:*
udp      0      0 localhost:domain      0.0.0.0:*
udp      0      0 andreyex.ru:ntp       0.0.0.0:*
```

Список только прослушивающихся портов UNIX с помощью netstat -lx

```
# netstat -lx
```

```
Active UNIX domain sockets (only servers)
Proto RefCnt Flags     Type      State      I-Node Path
unix  2 [ ACC ]     STREAM   LISTENING  19693  tmp/core.adm.internal
unix  2 [ ACC ]     SEQPACKET LISTENING  8723   /run/udev/control
unix  2 [ ACC ]     STREAM   LISTENING  12566  /var/run/dbus/system_bus_socket
unix  2 [ ACC ]     STREAM   LISTENING  16948  /var/run/fail2ban/fail2ban.sock
unix  2 [ ACC ]     STREAM   LISTENING  19702  tmp/core.sock
```

3. Показать статистику для каждого протокола

Показать статистику для всех портов с использованием netstat -s

```
# netstat -s
```

Ip:

```
190566 total packets received
0 forwarded
0 incoming packets discarded
189618 incoming packets delivered
170462 requests sent out
16 dropped because of missing route
```

Icmp:

```
74 ICMP messages received
0 input ICMP message failed.
ICMP input histogram:
destination unreachable: 22
echo requests: 52
```

....

ASTRA LINUX GUIDE

netstat

Показать статистику для TCP (или) UDP портов с использованием netstat -st (или) -su

```
# netstat -st
```

```
# netstat -su
```

4. PID и названий программ в выводе netstat с помощью команды netstat -p

Опция netstat -p может быть объединена с любым другим вариантом netstat. Это добавит «PID/Название программы» на выходе netstat. Это очень полезно при отладке, чтобы определить, какая программа работает на определенном порту.

```
# netstat -pt
```

Active Internet connections (w/o servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	localhost:56642	localhost:46846	TIME_WAIT	-
tcp	0	0	localhost:56642	localhost:46748	TIME_WAIT	-

5. Не разрешать хост, порт и имя пользователя в выводе netstat

Если вы не хотите отображать имя хоста, порт или пользователя, используйте netstat с опцией -n. Это будет отображаться в цифрах, и не разрешать имя хоста, имя порта, имя пользователя.

Это также ускоряет выход, так как netstat не выполняет какие-либо просмотров.

```
# netstat -an
```

Если вы не хотите один из этих трех пунктов (порт или хост, или пользователя), используйте следующие команды.

```
# netsat -a --numeric-ports
```

```
# netsat -a --numeric-hosts
```

```
# netsat -a --numeric-users
```

6. Непрерывная печать информации netstat

netstat будет печатать информацию непрерывно каждые несколько секунд.

```
# netstat -c
```

Active Internet connections (w/o servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	localhost:37840	localhost:webcache	ESTABLISHED

tcp	0	0	andreyex.ru:vlsi-lm	213.132.93.178:24080	ESTABLISHED
-----	---	---	---------------------	----------------------	-------------

tcp	0	0	localhost:56642	localhost:47258	TIME_WAIT
-----	---	---	-----------------	-----------------	-----------

tcp	0	0	localhost:56642	localhost:47150	TIME_WAIT
-----	---	---	-----------------	-----------------	-----------

^C

Выход из печати: Ctrl+C.

ASTRA LINUX GUIDE

netstat

7. Найти номера поддерживающей адрес семейств в вашей системе

```
netstat --verbose
```

8. Отображение информации о маршрутизации ядра с помощью netstat -r

```
# netstat -r
```

```
Kernel IP routing table
Destination     Gateway         Genmask        Flags MSS Window irtt Iface
default         gw.msk.ispsyste 0.0.0.0      UG    0 0        0 eth0
213.159.208.0  0.0.0.0        255.255.254.0 U     0 0        0 eth0
```

9. Узнайте, на каком порту работает программа

```
# netstat -ap | grep ssh
```

(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)

```
tcp    0  0 0.0.0.0:28456      0.0.0.0:*      LISTEN   779/sshd
tcp    0  0 andreyex.ru:28456  213.132.93.178:13430 ESTABLISHED 2893/sshd: andreyex
tcp    0  0 andreyex.ru:28456  213.132.93.178:13106 ESTABLISHED 2393/sshd: andreyex
tcp6   0  0 [::]:28456        [::]:*      LISTEN   779/sshd
```

Узнайте, какой процесс использует определенный порт:

```
# netstat -an | grep ':80'
```

10. Показать список сетевых интерфейсов

```
# netstat -i
```

```
Kernel Interface table
Iface      MTU RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1555 2765202 0 0 0 86602 0 0 0 BMRU
lo       65536 93149 0 0 0 93149 0 0 0 LRU
```

Отображение расширенной информации об интерфейсах (по аналогии с ifconfig) с использованием netstat -ie:

```
# netstat -ie
```

```
Kernel Interface table
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1550
        inet 213.159.209.228 netmask 255.255.254.0 broadcast 213.159.209.255
              inet6 fe80::5054:ff:fe80:19a4 prefixlen 64 scopeid 0x20<link>
        ether 52:54:00:80:19:a4 txqueuelen 1000 (Ethernet)
              RX packets 2772322 bytes 189451708 (180.6 MiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 86767 bytes 137897931 (131.5 MiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ss

Команда ss — это инструмент, который используется для отображения информации о сетевых сокетах в системе Linux. Инструмент отображает более подробную информацию, чем команда netstat, которая используется для отображения активных соединений сокетов.

1. Перечисление всех соединений

Базовая команда ss без каких-либо опций просто выводит список всех соединений независимо от состояния, в котором они находятся.

Если ни одна из опций не используется, ss отображает список открытых не слушающих сокетов (например, TCP/UNIX/UDP), которые установили соединение.

2. Список слушающих и не слушающих портов

Вы можете получить список как слушающих, так и не слушающих портов, используя опцию -a, как показано ниже.

```
ss -a
```

3. Список прослушивающих сокетов

Чтобы отобразить только сокеты прослушивания, используйте флаг -l:

```
ss -l
```

4. Список всех TCP соединений

Чтобы отобразить все соединения TCP, используйте параметр -t:

```
ss -t
```

5. Список всех слушающих TCP соединения

Для просмотра всех слушающих TCP-сокетов используйте комбинацию -lt:

```
ss -lt
```

6. Список всех UDP соединений

Для просмотра всех сокетов с UDP соединениями используйте параметр -ua:

```
ss -ua
```

7. Список всех слушающих UDP соединений

Для просмотра списка подключений UDP используйте параметр -lu.

```
ss -lu
```

8. Отображение у сокетов PID (идентификаторов процессов)

Для отображения идентификаторов процессов, связанных с соединениями сокетов, используйте флаг -p:

```
ss -p
```

9. Показать сводную статистику

Чтобы вывести сводную статистику, используйте опцию `-s`.

```
ss -s
```

10. Показать сокеты IPv4 и IPv6

Если вам интересны соединения через сокет IPv4, используйте опцию `-4`.

```
ss -4
```

Чтобы отобразить соединения IPv6, используйте параметр `-6`.

```
ss -6
```

11. Фильтр соединений по номеру порта

Команда `ss` также позволяет фильтровать номер порта сокета или номер адреса. Например, для отображения всех соединений сокетов с портом назначения или исходным портом `ssh` выполните команду.

```
ss -at '( dport = :22 or sport = :22 )'
```

Альтернативная команда:

```
ss -at '( dport = :ssh or sport = :ssh )'
```

12. Вывод номеров портов в числовом формате, а не имени в ss

По умолчанию команда `ss` показывает имена портов, чтобы выводились порты в виде чисел, используйте опцию `-n`:

```
ss -n
```

13. Поиск открытых портов на Linux

Следующая команда покажет все прослушиваемые порты для TCP и UDP соединений в виде цифровых значений:

```
sudo ss -lntu
```

14. Поиск программ, которые прослушивают порты на Linux

Если добавить ключ `-r`, то программа дополнительно покажет процессы, использующие сокет:

NMAP

nmap — это аббревиатура от «Network Mapper», на русский язык наиболее корректно можно перевести как «сетевой картограф». Возможно, это не лучший вариант перевода на русский язык, но он довольно точно отображает суть — инструмент для исследования сети и проверки безопасности. Утилита кроссплатформенна, бесплатна, поддерживаются операционных системы Linux, Windows, FreeBSD, OpenBSD, Solaris, Mac OS X.

Рассмотрим использование утилиты в Debian. В стандартной поставке дистрибутива nmap отсутствует, установим его командой

```
# aptitude install nmap
```

Nmap умеет сканировать различными методами — например, UDP, TCP connect(), TCP SYN (полуоткрытое), FTP proxy (прорыв через ftp), Reverse-ident, ICMP (ping), FIN, ACK, SYN и NULL-сканирование. Выбор варианта сканирования зависит от указанных ключей, вызов nmap выглядит следующим образом:

```
nmap <ключи> цель
```

Для опытов возьмем специальный хост для экспериментов, созданный самими разработчиками nmap — scanme.nmap.org. Выполним от root'a
Ключи сканирования задавать необязательно — в этом случае nmap проверит хост на наличие открытых портов и служб, которые слушают эти порты.
Запустим командой:

```
# nmap scanme.nmap.org
```

Через несколько секунд получим результат:
Interesting ports on scanme.nmap.org (74.207.244.221):
Not shown: 998 closed ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http

Ничего необычного, ssh на стандартном порту и http на 80. Nmap распознаёт следующие состояния портов: open, filtered, closed, или unfiltered. Open означает, что приложение на целевой машине готово для принятия пакетов на этот порт. Filtered означает, что брандмауэр, фильтр, или что-то другое в сети блокирует порт, так что Nmap не может определить, является ли порт открытым или закрытым. Closed — не связанны в данный момент ни с каким приложением, но могут быть открыты в любой момент. Unfiltered порты отвечают на запросы Nmap, но нельзя определить, являются ли они открытыми или закрытыми.

```
# nmap -O scanme.nmap.org
```

Хint: Если во время сканирования нажать пробел — можно увидеть текущий прогресс сканирования и на сколько процентов он выполнен. Через несколько секунд получаем ответ, в котором пока что интересна строчка Device type:

Device type: general purpose|WAP|webcam|storage-misc
Running (JUST GUESSING) : Linux 2.6.X|2.4.X (93%), AXIS Linux 2.6.X (89%), Linksys Linux 2.4.X (89%)
Aggressive OS guesses: Linux 2.6.17 - 2.6.28 (93%), Linux 2.6.9 - 2.6.27 (93%), Linux 2.6.24 (Fedora 8) (92%), Linux 2.6.18 (Slackware 11.0) (92%), Linux 2.6.19 - 2.6.26 (92%), OpenWrt (Linux 2.4.32) (91%), Linux 2.6.22 (91%), Linux 2.6.22 (Fedora Core 6) (90%), Linux 2.6.13 - 2.6.27 (90%), Linux 2.6.9 - 2.6.18 (90%)
No exact OS matches for host (test conditions non-ideal).

Вообще, точную версию ядра средствами nmap определить невозможно, но примерную дату «свежести» и саму операционную систему определить можно. Можно просканировать сразу несколько хостов, для этого надо их перечислить через пробел:

```
# nmap -O example.com example2.com
```

Вернемся к нашему подопытному хосту. Хочется узнать поподробнее, какой используется софт. Попробуем уточнить полученные данные с помощью ключей -sV:

```
# nmap -sV example.com example2.com
```

Получим ответ:

```
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 5.3p1 Debian 3ubuntu7 (protocol 2.0)
80/tcp open http Apache httpd 2.2.14 ((Ubuntu))
Service Info: OS: Linux
```

Прогресс налицо — мы узнали точные названия используемых служб и даже их версии, а заодно узнали точно, какая операционная система стоит на сервере. С расшифровкой никаких проблем не возникает, все вполне понятно.

Агрессивное сканирование можно провести, указав ключ -A

```
# nmap -A scanme.nmap.org
```

Nmap выведет очень много информации, я не стану приводить пример. Сканирование может длится довольно долго, занимая несколько минут.

В локальных сетях или просто имея на руках диапазон ip адресов, удобно проверить их занятость с помощью ключей -sP:

```
# nmap -sP 192.168.1.0/24
```

Сканирование проходит довольно быстро, так как по сути это обычный ping-тест, отвечает ли хост на ping. Следует учесть, что хост может не отвечать на ping из-за настроек фаерволла. Если нужный участок сети нельзя ограничить маской, можно указать диапазон адресов, с какого и по какой надо провести сканирование. Например, есть диапазон адресов с 192.168.1.2 до 192.168.1.5. Тогда выполним:

```
# nmap -sP 192.168.1.2-5
```

Ответ будет выглядеть так:

```
Host 192.168.1.2 is up (0.0023s latency)
Host 192.168.1.3 is up (0.0015s latency)
Host 192.168.1.4 is up (0.0018s latency)
Host 192.168.1.5 is up (0.0026s latency)
```

В моем случае все ip в данный момент были в сети.

Это далеко не все возможности nmap, но уместить их в рамках одной статьи несколько сложновато.

Если вам ближе GUI — есть замечательная утилита Zenmap — графическая оболочка для nmap, умеющая заодно и строить предполагаемую карту сети.

TELNET

Telnet - это сетевая утилита, которая позволяет соединиться с удаленным портом любого компьютера и установить интерактивный канал связи, например, для передачи команд или получения информации. Можно сказать, что это универсальный браузер в терминале, который умеет работать со множеством сетевых протоколов.

Эта утилита очень часто использовалась раньше, для удаленного управления компьютером с Linux, но потом ей на замену пришел защищенный протокол SSH. Но telnet все еще используется, например, для тестирования сети, проверки портов, а также для взаимодействия с различными IoT устройствами и роутерами.

ЧТО ТАКОЕ TELNET?

Как я уже сказал, эта утилита предназначена для создания интерактивного соединения между удаленными компьютерами. Она работает по протоколу TELNET, но этот протокол поддерживается многими сервисами, поэтому ее можно использовать для управления ими. Протокол работает на основе TCP, и позволяет передавать обычные строковые команды на другое устройство. Он может использоваться не только для ручного управления но и для взаимодействия между процессами.

Для работы с этим протоколом мы будем использовать утилиту telnet, ею очень просто пользоваться. Давайте рассмотрим синтаксис telnet:

```
$ telnet опции хост порт
```

Хост - это домен удаленного компьютера, к которому следует подключиться, а порт - порт на этом компьютере. А теперь давайте рассмотрим основные опции:

- 4 - принудительно использовать адреса ipv4;
- 6 - принудительно использовать адреса ipv6;
- 8 - использовать 8-битную кодировку, например, Unicode;
- E - отключить поддержку Escape последовательностей;
- a - автоматический вход, берет имя пользователя из переменной окружения USER;
- b - использовать локальный сокет;
- d - включить режим отладки;
- r - режим эмуляции rlogin;
- e - задать символ начала Escape последовательности;
- l - пользователь для авторизации на удаленной машине.

Это все, что касается команды telnet для установки соединения. Но соединение с удаленным хостом, это только полдела. После установки подключения telnet может работать в двух режимах:

Построчный - это предпочтительный режим, здесь строка текста редактируется на локальном компьютере и отправляется только тогда, когда она будет полностью готова. На такая возможность есть не всегда и не у всех сервисов;

Посимвольный - все набираемые вами символы отправляются на удаленный сервер. Тут будет сложно что-либо исправить, если вы допустили ошибку, потому что Backspace тоже будет отправляться в виде символа и стрелки движения тоже.

Использование telnet заключается в передаче специальных команд. У каждого сервиса свои команды, но у протокола есть свои команды telnet, которые можно применять в консоли telnet.

CLOSE - закрыть соединение с сервером;
ENCRYPT - шифровать все передаваемые данные;
LOGOUT - выйти и закрыть соединение;
MODE - переключить режим, со строчного на символьный или с символьного на строчный;
STATUS - посмотреть статус соединения;
SEND - отправить один из специальных символов telnet;
SET - установить значение параметра;
OPEN - установить подключение через telnet с удаленным узлом;
DISPLAY - отобразить используемые спецсимволы;
SLC - изменить используемые спецсимволы.

Мы не будем рассматривать все команды, поскольку они вам вряд ли понадобятся, а если и понадобятся, то вы легко сможете их найти в официальной документации.

КАК ПОЛЬЗОВАТЬСЯ TELNET?

Дальше мы рассмотрим как использовать telnet для решения ваших задач. Обычно, утилита уже установлена в большинстве систем, но если это не так, то вы можете установить telnet из официальных репозиториев, например, в Ubuntu:

```
$ sudo apt install telnet
```

Теперь перейдем к применению утилиты. Изначально она использовалась для удаленного управления компьютером, но поскольку потом был разработан более безопасный протокол SSH, использовать ее перестали.

1. ДОСТУПНОСТЬ СЕРВЕРА

Утилита все еще может быть полезной при проверке доступности узла, для этого просто передайте ей ip адрес или имя хоста:

```
$ telnet 192.168.1.243
```

Для этого не обязательно применять telnet, есть ping.

2. ПРОВЕРКА ПОРТА

С помощью telnet мы можем проверить доступность порта на узле, а это уже может быть очень полезным. Чтобы проверить порт telnet выполните:

```
$ telnet localhost 123  
$ telnet localhost 22
```

В первом случае мы видим, что соединение никто не принимает, во втором же выводится сообщение об успешном подключении и приветствие SSH сервера.

3. ОТЛАДКА

Чтобы включить режим отладки и выводить более подробную информацию во время работы используйте опцию -d во время подключения:

```
sudo telnet -d localhost 22
```

4. КОНСОЛЬ TELNET

Использование консоли telnet тоже важный момент в разборе как пользоваться telnet. В основном режиме вы можете выполнять команды, на удаленном сервере, если же вы хотите адресовать команду именно telnet, например, для настройки ее работы, необходимо использовать спецсимвол для открытия консоли, обычно утилита сразу говорит вам что это за символ, например, по умолчанию используется "^[":

Для его активации вам нужно нажать сочетание клавиш Ctrl+[, затем вы увидите приглашение ввода telnet.

Чтобы посмотреть все доступные команды, вы можете набрать ?. Например, вы можете посмотреть статус подключения:

```
telnet> status
```

Здесь есть и другие интересные возможности. Такие вещи можно проделывать при любом подключении с помощью утилиты telnet.

5. ПОСМОТРЕТЬ САЙТ TELNET

Один из распространенных способов использования telnet - это тестирование сайта из консоли. Да, красивую веб-страницу вы не получите, но можете вручную собрать запросы и видеть все переданные сервером данные.

```
telnet opennet.ru 80
```

Затем наберите команду веб-серверу:

```
GET /
```

Веб сервер вернет полностью страницу, а также заголовки, которые необходимы для ее отображения браузером.

6. УДАЛЕННОЕ УПРАВЛЕНИЕ TELNET

Настоятельно не рекомендуется использовать небезопасный telnet для удаленного управления, потому что все команды и пароли могут быть прослушаны сторонним пользователем. Но иногда, например, для роутеров telnet все же используется для удаленного управления. Все работает точно так же, как и для других подключений, только нужно использовать порт 23, а на удаленном компьютере должен быть установлен telnet-server:

```
telnet localhost 23
```

Тут порт можно даже не указывать, потому что по умолчанию будет использоваться именно 23. Далее, вам нужно ввести логин и пароль, а затем вы сможете выполнять команды в удаленной системе.