14 tcpdump Commands to Capture Network Traffic in Linux

April 13, 2018LINUX COMMANDS, LINUX HOWTOUpdated April 13, 2018

tcpdump is a network troubleshooting command which is also known as a packet sniffer is used to capture and display packets from a network. tcpdump allows users to capture and display TCP/IP and other packets being transmitted or received over the network to which computer is attached. It allows to search the output in real time by redirecting the captured data to grep or the data can be written to a file for later searching. You can apply filters to the packets and can avoid the traffic which you do not want to see. To run tcpdump command you requires root access.

You can capture all the data going across your local network and put that data in a file. This does not display real-time output to the screen but captures the packets on the network to the file indicated. You can save the whole packets or part of the packets (header). You can choose to capture and analyze the traffic over one network interface or all the network interface. In this article, we will learn how to use tcpdump command to analyze the traffic flowing on a Linux machine.

1) Analyse traffic on all the interface

tcpdump is not installed by default, so you need to install it before

On Ubuntu 16.04

```
# apt install tcpdump
```

On Centos 7

```
# yum install tcpdump
```

When you use tcpdump without any option, it will analyze the traffic on all the interface.

```
# tcpdump
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

07:41:25.886307 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 1435074392:1435074508, a

ck 4135933864, win 381, options [nop,nop,TS val 3387567505 ecr 18
335689], length 116

07:41:25.886932 IP li339-47.members.linode.com.49063 > resolver08
.dallas.linode.com.domain: 61296+ PTR? 5.7.255.169.in-addr.arpa.
(42)

07:41:26.133811 IP 169.255.7.5.44284 > li339-47.members.linode.co
m.ssh: Flags [.], ack 0, win 722, options [nop,nop,TS val 1833575
7 ecr 3387567484], length 0

07:41:26.133851 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 116:232, ack 1, win 381, options [nop,nop,
TS val 3387567753 ecr 18335757], length 116

07:41:26.142929 IP resolver08.dallas.linode.com.domain > li339-47
.members.linode.com.49063: 61296 NXDomain 0/0/0 (42)

.....

.....

07:41:26.680521 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 2724:3132, ack 1, win 381, options [nop,no
p,TS val 3387568299 ecr 18335894], length 408

^C

17 packets captured

18 packets received by filter

0 packets dropped by kernel
```

The format of the source is *source.port* where the source can be the hostname or the IP address. You can see on the first line, the packet captured at the timestamp *07:41:25.886307* is an IP protocol which is originated at hostname *li339-47.members.linode.com* and port *ssh,* that is

why you see *li339-47.members.linode.com.ssh.* The packet is destined for *169.255.7.5.44284* ack flag.

You need to hit cancel button in order to stop it. If you need more information, tcpdump provides several options that enhance or modify its output:

- `-i interface` : Listen on the specified interface.
- `-n` : Don't resolve hostnames. You can use **-nn** to don't resolve hostnames or port names.
- `-t` : Don't print a timestamp on each dump line.
- `-X` : Show the packet's contents in both hex and ascii.
- `-v, -vv, -vvv` : Increase the amount of packet information you get back.
- `-c N` : Only get N number of packets and then stop.
- `-s` : Define the snaplen (size) of the capture in bytes. Use -s0 to get everything, unless you are intentionally capturing less.
- `-S` : Print absolute sequence numbers.
- `-q` : Show less protocol information.
- `-w file` : Write the raw packets to file rather

2) List available interfaces

You can list the available interface with the `-D` parameter

```
# tcpdump -D

1.eth0

2.nflog (Linux netfilter log (NFLOG) interface)

3.nfqueue (Linux netfilter queue (NFQUEUE) interface)

4.any (Pseudo-device that captures on all interfaces)

5.lo [Loopback]
```

With that, you can choose which interface to use.

3) Analyze a specific interface and limit packets

Now you can decide to analyze the traffic on a specified interface with the `-i` parameter and limit the number of packets to capture with `-c`

```
# tcpdump -i eth0 -c 5

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

08:57:09.186418 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 1435431156:1435431272, ack 4135945080, win
419, options [nop,nop,TS val 3392110805 ecr 19471515], length 116

08:57:09.186855 IP li339-47.members.linode.com.33326 > resolver08
.dallas.linode.com.domain: 9787+ PTR? 5.7.255.169.in-addr.arpa. (
42)

08:57:09.335228 IP 134.119.220.87.45873 > li339-47.members.linode
.com.60342: Flags [S], seq 3684168813, win 1024, length 0

08:57:09.335264 IP li339-47.members.linode.com.60342 > 134.119.22
0.87.45873: Flags [R.], seq 0, ack 3684168814, win 0, length 0

08:57:09.378999 IP 134.119.220.87.45873 > li339-47.members.linode
.com.25070: Flags [S], seq 3509221600, win 1024, length 0

5 packets captured

13 packets received by filter

0 packets dropped by kernel
```

4) Capture data using ip address and port

As you can see on the capture above, we don't have the port number and the IP address of the source. You can use the `-nn` in order to have it

```
# tcpdump -i eth0 -c 5 -nn

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes
```

```
09:17:09.572425 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 1435457792:1435457908, ack 4135947356, win 419, options [
nop,nop,TS val 3393311191 ecr 19771613], length 116

09:17:09.605048 IP 96.126.114.47.32887 > 204.11.201.10.123: NTPv4
, Client, length 48

09:17:09.663754 IP 204.11.201.10.123 > 96.126.114.47.32887: NTPv4
, Server, length 48

09:17:09.785600 IP 169.255.7.5.44284 > 96.126.114.47.22: Flags [.
], ack 0, win 722, options [nop,nop,TS val 19771669 ecr 339331118
3], length 0

09:17:09.785646 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 116:700, ack 1, win 419, options [nop,nop,TS val 33933114
04 ecr 19771669], length 584

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

5) Intercept packet from a specific port

You can decide to intercept packets to a specified port number with `port` parameter.

```
# tcpdump -i eth0 -c 5 -nn port 22

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

09:27:27.773270 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 1435459900:1435460016, ack 4135948192, win 419, options [
nop,nop,TS val 3393929392 ecr 19926162], length 116

09:27:27.773357 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 116:232, ack 1, win 419, options [nop,nop,TS val 33939293
92 ecr 19926162], length 116
```

```
09:27:28.032620 IP 169.255.7.5.44284 > 96.126.114.47.22: Flags [.
], ack 0, win 722, options [nop,nop,TS val 19926230 ecr 339392938
4], length 0

09:27:28.032655 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 232:648, ack 1, win 419, options [nop,nop,TS val 33939296
52 ecr 19926230], length 416

09:27:28.032668 IP 169.255.7.5.44284 > 96.126.114.47.22: Flags [.
], ack 116, win 722, options [nop,nop,TS val 19926230 ecr 3393929
392], length 0

5 packets captured

6 packets received by filter

0 packets dropped by kernel
```

6) Intercept packet by ignoring a specific port

You can decide to ignore a port when you intercept packets. This is possible with the `not port` parameter

```
# tcpdump -i eth0 -c 5 -nn not port 22

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

11:15:53.784094 IP 134.119.220.87.45873 > 96.126.114.47.32724: Fl
ags [S], seq 1210911834, win 1024, length 0

11:15:53.784139 IP 96.126.114.47.32724 > 134.119.220.87.45873: Fl
ags [R.], seq 0, ack 1210911835, win 0, length 0

11:15:53.910633 IP 134.119.220.87.45873 > 96.126.114.47.32724: Fl
ags [R], seq 1210911835, win 1200, length 0

11:15:53.911319 IP 134.119.220.87 > 96.126.114.47: ICMP host 134.
119.220.87 unreachable - admin prohibited, length 48
```

```
11:15:56.327699 IP 134.119.220.87.45873 > 96.126.114.47.18566: Fl
ags [S], seq 3213454109, win 1024, length 0

5 packets captured

6 packets received by filter

0 packets dropped by kernel
```

7) Intercept packets from a specific protocol

You can decide to capture icmp or tcp packets only

```
# tcpdump -i eth0 -c 5 -nn tcp

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

09:49:33.371487 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 1435550388:1435550504, ack 4135954104, win 438, options [
nop,nop,TS val 3395254990 ecr 20257561], length 116

09:49:33.371612 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 116:232, ack 1, win 438, options [nop,nop,TS val 33952549
90 ecr 20257561], length 116

09:49:33.371788 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 232:452, ack 1, win 438, options [nop,nop,TS val 33952549
91 ecr 20257561], length 220

09:49:33.371956 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 452:648, ack 1, win 438, options [nop,nop,TS val 33952549
91 ecr 20257561], length 196

09:49:33.631626 IP 169.255.7.5.44284 > 96.126.114.47.22: Flags [.
], ack 116, win 722, options [nop,nop,TS val 20257629 ecr 3395254
981], length 0

5 packets captured

7 packets received by filter
```

```
0 packets dropped by kernel
```

You can just replace tcp by icmp for that one

8) Record log to some specific file

It is possible to save the captured packets in a file. By default, when capturing packets into a file, it will save only 68 bytes of the data from the each packet. Rest of the information is ignored. You can use `-s` to tell tcpdump how many bytes for each packets to save and specify 0 as packets snapshot length tells tcpdump to save whole packet.

```
# tcpdump -i eth0 -c 5 -nn tcp -w packets-record.cap -s 0

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

9) Read tcpdump record file

You can't read the content of a file which saves tcpdump packets with the common commands such as cat or less but you need to use the `-r` parameter of the tcpdump command

```
# tcpdump -r packets-record.cap

reading from file packets-record.cap, link-type EN10MB (Ethernet)

10:06:25.310077 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 1435573932:1435573976, ack 4135958592, win
457, options [nop,nop,TS val 3396266929 ecr 20510549], length 44

10:06:25.565590 IP 169.255.7.5.44284 > li339-47.members.linode.co
m.ssh: Flags [.], ack 0, win 722, options [nop,nop,TS val 2051061
6 ecr 3396266919], length 0

10:06:25.565633 IP li339-47.members.linode.com.ssh > 169.255.7.5.
44284: Flags [P.], seq 44:160, ack 1, win 457, options [nop,nop,T
S val 3396267184 ecr 20510616], length 116
```

```
10:06:25.570384 IP 169.255.7.5.44284 > li339-47.members.linode.co
m.ssh: Flags [.], ack 44, win 722, options [nop,nop,TS val 205106
17 ecr 3396266929], length 0

10:06:25.827438 IP 169.255.7.5.44284 > li339-47.members.linode.co
m.ssh: Flags [.], ack 160, win 722, options [nop,nop,TS val 20510
681 ecr 3396267184], length 0
```

10) Capture packets with more information

You can scan the network more deeply. You can use the combinations of the command to filter what you need

```
# tcpdump -i eth0 -c 5 -ttttnnvvS

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes

2018-04-10 10:32:36.073756 IP (tos 0x10, ttl 64, id 14601, offset
0, flags [DF], proto TCP (6), length 96)

    96.126.114.47.22 > 169.255.7.5.44284: Flags [P.], cksum 0x840
4 (incorrect -> 0x570b), seq 1435611412:1435611456, ack 413596947
2, win 495, options [nop,nop,TS val 3397837693 ecr 20903238], len
gth 44

2018-04-10 10:32:36.073896 IP (tos 0x10, ttl 64, id 14602, offset
0, flags [DF], proto TCP (6), length 168)

    96.126.114.47.22 > 169.255.7.5.44284: Flags [P.], cksum 0x844
c (incorrect -> 0x14ec), seq 1435611456:1435611572, ack 413596947
2, win 495, options [nop,nop,TS val 3397837693 ecr 20903238], len
gth 116

2018-04-10 10:32:36.074118 IP (tos 0x10, ttl 64, id 14603, offset
0, flags [DF], proto TCP (6), length 200)

    96.126.114.47.22 > 169.255.7.5.44284: Flags [P.], cksum 0x846
c (incorrect -> 0x52d8), seq 1435611572:1435611720, ack 413596947
2, win 495, options [nop,nop,TS val 3397837693 ecr 20903238], len
gth 148

2018-04-10 10:32:36.083469 IP (tos 0x8, ttl 53, id 26190, offset
0, flags [none], proto ICMP (1), length 68)
```

```
    134.119.220.87 > 96.126.114.47: ICMP host 134.119.220.87 unre
achable - admin prohibited, length 48

        IP (tos 0x28, ttl 48, id 23212, offset 0, flags [DF], pro
to TCP (6), length 40)

    96.126.114.47.47317 > 134.119.220.87.45873: Flags [R.], cksum
0x5362 (correct), seq 0, ack 96384300, win 0, length 0

2018-04-10 10:32:36.084338 IP (tos 0x0, ttl 244, id 32726, offset
0, flags [none], proto TCP (6), length 40)

    134.119.220.87.45873 > 96.126.114.47.47317: Flags [R], cksum
0x4ec2 (correct), seq 96384300, win 1200, length 0

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

11) Capture packets coming from a remote host

To only show packets which come from a specific IP, you use the `src` parameter

```
# tcpdump -i eth0 -c 5 -ttttnnvvS src host 96.126.114.1

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes

2018-04-10 11:27:28.498964 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

2018-04-10 11:28:08.614258 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

2018-04-10 11:28:53.621982 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

2018-04-10 11:29:33.511165 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46
```

```
2018-04-10 11:30:13.837251 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

You can see the requests packets

12) Capture packets in the destination to a remote host

It is possible to show only packages that have a specific destination. For example, you can show packages in the destination to the router

```
# tcpdump -i eth0 -c 5 -ttttnnvvS dst host 96.126.114.1

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes

2018-04-10 11:34:15.107495 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:35:00.547492 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:35:47.907837 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:36:12.867576 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:36:39.534063 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

Here you can see the replies packets

13) Capture both incoming and outgoing packets of a specific host

In the two commands above, we used `src` and `dst` to capture incoming and outgoing packets from a specific host in two different times. But it is possible to do it directly in one command with only the `host` parameter

```
# tcpdump -i eth0 -c 5 -ttttnnvvS host 96.126.114.1

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes

2018-04-10 11:37:49.720992 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:37:49.725683 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

2018-04-10 11:38:14.894130 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

2018-04-10 11:38:14.900008 ARP, Ethernet (len 6), IPv4 (len 4), R
eply 96.126.114.1 is-at 00:00:0c:9f:f0:06, length 46

2018-04-10 11:38:39.854051 ARP, Ethernet (len 6), IPv4 (len 4), R
equest who-has 96.126.114.1 tell 96.126.114.47, length 28

5 packets captured

5 packets received by filter

0 packets dropped by kernel
```

Now you can see both request and reply packets.

14) Capture packet using port range

It is possible to use a range of ports in order to capture your network traffic.

```
# tcpdump -i eth0 -c 3 -nns 0 portrange 20-23

tcpdump: verbose output suppressed, use -v or -vv for full protoc
ol decode
```

```
listening on eth0, link-type EN10MB (Ethernet), capture size 2621
44 bytes

11:59:45.996312 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 1435738516:1435738632, ack 4136021820, win 875, options [
nop,nop,TS val 3403067615 ecr 22210718], length 116

11:59:45.996512 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 116:232, ack 1, win 875, options [nop,nop,TS val 34030676
15 ecr 22210718], length 116

11:59:45.996728 IP 96.126.114.47.22 > 169.255.7.5.44284: Flags [P
.], seq 232:452, ack 1, win 875, options [nop,nop,TS val 34030676
16 ecr 22210718], length 220

3 packets captured

5 packets received by filter

0 packets dropped by kernel
```

Conclusion

Tcpdump offers some options to filter the packets of the traffic to capture. Although packet sniffers are useful diagnostic tools, they can also be abused. For instance, unscrupulous individuals can run packet sniffers to capture passwords that others send over the network. Depending on your network configuration, this trick can work even if the packet sniffer isn't running on either the sending or the receiving computer. For this reason, many organizations have policies forbidding the use of packet sniffers except under limited circumstances.

Linux Commands, Monitoring, tcpdump Command

About Alain Francois

IT Linux administrator passionate of free and open source software, I work on Linux Systems since some years on installations and deployments of certain solutions. I like to share my experiences with a wider audience by training and conferences.