

TSHARK

Description

TShark is a network protocol analyzer.

It lets you capture packet data from a live network, or read packets from a previously saved capture file, either printing a decoded form of those packets to the standard output or writing the packets to a file. TShark's native capture file format is pcapng format, which is also the format used by Wireshark and various other tools.

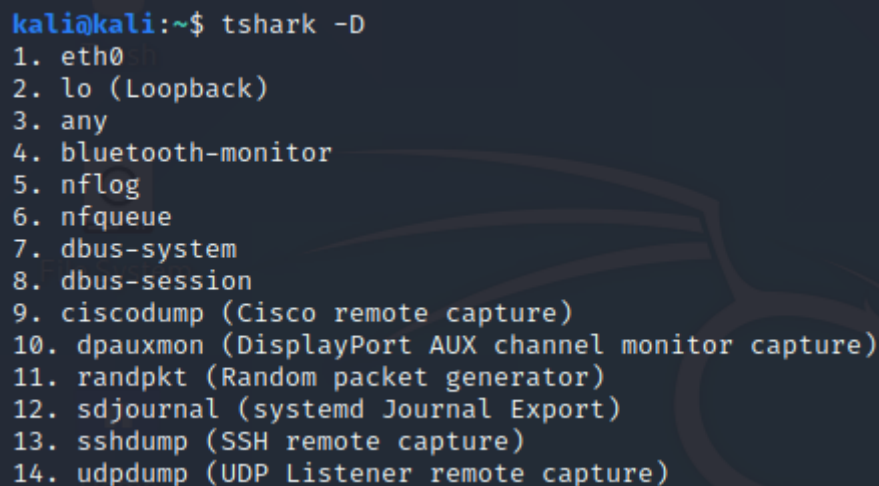
Without any options set, TShark will work much like tcpdump.

It will use the pcap library to capture traffic from the first available network interface and displays a summary line on the standard output for each received packet.

Usage Example

To start we can list the interfaces whose traffic tshark can capture with the command:

```
kali@kali:~$ tshark -D
```



```
kali@kali:~$ tshark -D
1. eth0
2. lo (Loopback)
3. any
4. bluetooth-monitor
5. nflog
6. nfqueue
7. dbus-system
8. dbus-session
9. ciscodump (Cisco remote capture)
10. dpauxmon (DisplayPort AUX channel monitor capture)
11. randpkt (Random packet generator)
12. sdjournal (systemd Journal Export)
13. sshdump (SSH remote capture)
14. udpdump (UDP Listener remote capture)
```

We have a long list of interfaces. But for our demonstration we will use the "eth0" interface.

Well before launching the capture we first generate traffic on the interface with a ping to *google.fr*.

kali@kali:~\$ ping google.fr

```
kali@kali:~$ ping google.fr
PING google.fr (216.58.215.35) 56(84) bytes of data.
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=1 ttl=118 time=15.2 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=2 ttl=118 time=11.2 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=3 ttl=118 time=22.1 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=4 ttl=118 time=12.2 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=5 ttl=118 time=13.8 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=6 ttl=118 time=20.0 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=7 ttl=118 time=57.2 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=8 ttl=118 time=12.3 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=9 ttl=118 time=10.9 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=10 ttl=118 time=10.4 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=11 ttl=118 time=14.0 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=12 ttl=118 time=14.3 ms
 64 bytes from par21s17-in-f3.1e100.net (216.58.215.35): icmp_seq=13 ttl=118 time=11.0 ms
```

To start capturing traffic on this interface we will use the command

kali@kali:~\$ tshark -i eth0

```
kali@kali:~$ sudo tshark -i eth0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
 1 0.000000000 10.0.2.15 → 8.8.8.8      DNS 69 Standard query 0x6435 A google.fr
 2 0.000071532 10.0.2.15 → 8.8.8.8      DNS 69 Standard query 0xca30 AAAA google.fr
 3 5.001394014 10.0.2.15 → 172.16.0.1   DNS 69 Standard query 0x6435 A google.fr
 4 5.001497730 10.0.2.15 → 172.16.0.1   DNS 69 Standard query 0xca30 AAAA google.fr
 5 5.007683029 172.16.0.1 → 10.0.2.15   DNS 85 Standard query response 0x6435 A google.f
r A 216.58.215.35
 6 5.021969482 172.16.0.1 → 10.0.2.15   DNS 97 Standard query response 0xca30 AAAA googl
e.fr AAAA 2a00:1450:4007:808::2003
 7 5.023500284 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x04ec, seq=1/2
56, ttl=64
 8 5.038621633 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x04ec, seq=1/2
56, ttl=118 (request in 7)
 9 5.039094213 10.0.2.15 → 172.16.0.1   DNS 86 Standard query 0x9c5c PTR 35.215.58.216.i
n-addr.arpa
10 5.048939856 172.16.0.1 → 10.0.2.15   DNS 124 Standard query response 0x9c5c PTR 35.21
5.58.216.in-addr.arpa PTR par21s17-in-f3.1e100.net
```

We can also limit the number of packets we will intercept with the `-c` option.

kali@kali:~\$ tshark -i eth0 -c 15

```
kali@kali:~$ sudo tshark -i eth0 -c 15
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
 1 0.000000000 10.0.2.15 → 172.16.0.1   DNS 69 Standard query 0x9af9 A google.fr
 2 0.000034373 10.0.2.15 → 172.16.0.1   DNS 69 Standard query 0x08fd AAAA google.fr
 3 0.022717625 172.16.0.1 → 10.0.2.15   DNS 97 Standard query response 0x08fd AAAA googl
e.fr AAAA 2a00:1450:4007:808::2003
 4 0.026314383 172.16.0.1 → 10.0.2.15   DNS 85 Standard query response 0x9af9 A google.f
r A 216.58.215.35
 5 0.026629022 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=1/2
56, ttl=64
```

We can also get more details about the different packages with the -V option.

```
kali@kali:~$ sudo tshark -i eth0 -c 15 -V
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface eth0, id 0
  Interface id: 0 (eth0)
    Interface name: eth0
    Encapsulation type: Ethernet (1)
    Arrival Time: Jan 22, 2022 08:13:50.401406760 EST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1642857230.401406760 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 98 bytes (784 bits)
    Capture Length: 98 bytes (784 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:icmp:data]
    Ethernet II, Src: PcsCompu_5c:65:26 (08:00:27:5c:65:26), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
      Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
        Address: RealtekU_12:35:02 (52:54:00:12:35:02)
          .... ..1. .... .... .... = LG bit: Locally administered address (this is NOT the factory default)
```

We also have the possibility to write the output of the command to a pcap file. To do this we use the -w option

```
kali@kali:~$ sudo tshark -i eth0 -c 15 -w tshark/test.pcap
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
15
```

To identify the ip we used the command nslookup google.fr and for the filter applied with tshark we used the command

```
kali@kali:~$ nslookup google.fr
Server:          172.16.0.1
Address:         172.16.0.1#53

Non-authoritative answer:
Name:   google.fr
Address: 216.58.215.35
```

kali@kali:~\$ tshark -i eth0 -c 15 host 216.58.215.35 and port 443

'host' is used to identify the host ip address and 'port' is used for the port .

How to analyze a .pcap file with tshark?

To start with we have to use the -r option followed by the name of the .pcap file to read it, which gives

```
kali@kali:~$ tshark -r test.pcap
```

(test.pcap is the file to be analyzed.)

```
7/54531, ttl=64
kali@kali:~$ tshark -r tshark/test.pcap
 1 0.000000000 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=251
0/52745, ttl=64
 2 0.011096487 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x054b, seq=251
0/52745, ttl=118 (request in 1)
 3 1.001772003 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=251
1/53001, ttl=64
 4 1.011836615 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x054b, seq=251
1/53001, ttl=118 (request in 3)
 5 2.004156849 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=251
2/53257, ttl=64
 6 2.015503709 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x054b, seq=251
2/53257, ttl=118 (request in 5)
 7 3.006016688 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=251
3/53513, ttl=64
 8 3.017083533 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x054b, seq=251
3/53513, ttl=118 (request in 7)
 9 4.008747035 10.0.2.15 → 216.58.215.35 ICMP 98 Echo (ping) request id=0x054b, seq=251
4/53769, ttl=64
10 4.030106788 216.58.215.35 → 10.0.2.15 ICMP 98 Echo (ping) reply id=0x054b, seq=251
```

The -T option is used to generate data in different formats, this can be very useful when you need a specific format for your analysis. In our example we will do it in json. The final command in this case is

```
kali@kali:~$ tshark -r test.pcap -T json
```

```
{,
{
  "_index": "packets-2022-01-22",
  "_type": "doc",
  "_score": null,
  "_source": {
    "layers": {
      "frame": {
        "frame.interface_id": "0",
        "frame.interface_id_tree": {
          "frame.interface_name": "eth0"
        },
        "frame.encap_type": "1",
        "frame.time": "Jan 22, 2022 08:54:22.916474136 EST",
        "frame.offset_shift": "0.000000000",
        "frame.time_epoch": "1642859662.916474136",
        "frame.time_delta": "0.990512979",
        "frame.time_delta_displayed": "0.990512979",
        "frame.time_relative": "3.006016688",
        "frame.number": "7",
        "frame.len": "98",
        "frame.cap_len": "98",
        "frame.marked": "0",
        "frame.ignored": "0",
        "frame.protocols": "eth:ethertype:ip:icmp:data"
```

In the following example we will extract the packet number with `-e frame.number` , the source ip address with `-e ip.src` and the destination ip address with `-e ip.dst`, which gives

```
kali@kali:~$ tshark -r test.pcap -T fields -e frame.number -e ip.src -e ip.dst
```

```
kali@kali:~$ tshark -r tshark/test.pcap -T fields -e frame.number -e ip.src -e ip.dst
1      10.0.2.15      216.58.215.35
2      216.58.215.35  10.0.2.15
3      10.0.2.15      216.58.215.35
4      216.58.215.35  10.0.2.15
5      10.0.2.15      216.58.215.35
6      216.58.215.35  10.0.2.15
7      10.0.2.15      216.58.215.35
8      216.58.215.35  10.0.2.15
9      10.0.2.15      216.58.215.35
10     216.58.215.35  10.0.2.15
11     10.0.2.15      216.58.215.35
12     216.58.215.35  10.0.2.15
13     10.0.2.15      216.58.215.35
14     216.58.215.35  10.0.2.15
15     10.0.2.15      216.58.215.35
kali@kali:~$
```

Conclusion

There is a lot to say about this tool, but I hope that this introduction to tshark has made your mouth water and that it will be useful.