
Getting network routing information

The tools in this category can be used to get the network routing information of a target. We will describe several tools that are commonly used for this purpose. Knowledge of the network routing information will allow the penetration tester to understand the network of the target machine, such as which path is taken by the packets sent from the penetration tester machine to the target machine. The routing information will also give a clue as to whether the particular target is protected by firewall.

Let us see the several tools that can help you get routing information.

tcptraceroute

The `tcptraceroute` tool can be used as a complement to the `traceroute` command. The `traceroute` command sends a UDP or ICMP echo request packet with a **Time To Live** (**TTL**) of one and increments the TTL until the packet reaches the target, while the `tcptraceroute` tool uses TCP SYN to send out the packet to the target.

The advantage of using `tcptraceroute` is that, nowadays, it is common to find a firewall device filtered `traceroute` packet, so it will not be possible to trace the network path to the target completely. However, this firewall still allows a packet to reach a particular TCP port in the target machine. By using `tcptraceroute`, we will be able to find the network path to the target, even though there is a firewall in front of it.

The `tcptraceroute` tool will receive a SYN/ACK packet if the port is open and a RST packet if the port is closed.

To access `tcptraceroute`, you can use the console and type the following command:

```
# tcptraceroute
```

This will display usage information on your screen.

Let's go for some action.

We run the `traceroute` command to trace our network route to the `example.com` domain as follows:

```
# traceroute www.example.com
```

The redacted result for this command is as follows:

```
traceroute to www. example .com (192.168.10.100), 30 hops max, 40 byte
packets
 1  192.168.1.1 (192.168.1.1)  8.382 ms  12.681 ms  24.169 ms
 2  1.static.192.168.xx.xx.isp (192.168.2.1)  47.276 ms  61.215 ms  61.057
ms
 3  * * *
 4  74.subnet192.168.xx.xx.isp (192.168.4.1)  68.794 ms  76.895 ms  94.154
ms
 5  isp2 (192.168.5.1)  122.919 ms  124.968 ms  132.380 ms
...
15  * * *
...
30  * * *
```

After route number `15`, we are no longer able to get the route information. Usually, this is because the `traceroute` packets are blocked by a filtering device.

We will try again using `tcptraceroute`, and we know that the targethost has an open TCP port for the web server (`80`). We can use the following command:

```
# tcptraceroute www.example.com
```

The result for this command is as follows:

```
Selected device eth0, address 192.168.1.107, port 41884 for outgoing
packets
Tracing the path to www.  example  .com (192.168.10.100) on TCP port 80
(www),                 30 hops max
 1  192.168.1.1  55.332 ms  6.087 ms  3.256 ms
 2  1.static.192.168.xx.xx.isp (192.168.2.1)    66.497 ms
50.436                 ms  85.326 ms
 3  * * *
 4  74.subnet192.168.xx.xx.isp (192.168.4.1)  56.252 ms  28.041 ms  34.607
ms
 5  isp2 (192.168.5.1)  51.160 ms  54.382 ms  150.168 ms
 6  192.168.6.1  106.216 ms  105.319 ms  130.462 ms
 7  192.168.7.1  140.752 ms  254.555 ms  106.610 ms
...
14  192.168.14.1  453.829 ms  404.907 ms  420.745 ms
15  192.168.15.1 615.886 ms  474.649 ms  432.609 ms
16  192.168.16.1 [open]  521.673 ms  474.778 ms  820.607 ms
```

This time, our packet is able to reach the targethost, and it gives us all the route information from our machine to the targethost.

tctrace

Another tool that can be used to do route analysis is `tctrace`. It works by sending a TCP SYN packet to the target.

To access `tctrace`, you can use the console and type the following command:

```
# tctrace -i<device> -d<targethost>
```

In the preceding command, `-i` is the network interface to the target and `-d` is the target.

To run `tctrace` to a target, the following command is used:

```
# tctrace -i eth0 -d www.example.com
```

The following result is obtained:

```
 1(1)   [192.168.1.1]
 2(1)   [192.168.2.1]
3(all)  Timeout
 4(3)   [192.168.4.1]
 5(1)   [192.168.5.1]
 6(1)   [192.168.6.1]
 7(1)   [192.168.7.1]
...
14(1)   [192.168.14.1]
15(1)   [192.168.15.1]
16(1)   [192.168.16.1] (reached; open)
```