NMAP

- Nmap is short for Network Mapper.
- Nmap is the most famous scanning tool used by penetration testers.
- It is an open-source Linux command-line tool that is used to scan IP addresses and ports in a network and to detect installed applications.

Why use Nmap?

- Nmap helps you to quickly map out a network without sophisticated commands or configurations.
- It also supports simple commands (for example, to check if a host is up) and complex scripting through the Nmap scripting engine.
- Ability to quickly recognize all the devices including servers, routers, switches, mobile devices, etc on single or multiple networks.
- Helps identify services running on a system including web servers, DNS servers, and other common applications. Nmap can also detect application versions with reasonable accuracy to help detect existing vulnerabilities.
- Nmap can find information about the operating system running on devices.
- During security auditing and vulnerability scanning, you can use Nmap to attack systems using existing scripts from the Nmap Scripting Engine.

Some Basic Scans

- **Ping scan —** Scans the list of devices up and running on a given subnet.

```
nmap -sn 192.168.1.1/24
```

- **Scan a single host —** Scans a single host for 1000 well-known ports. These ports are the ones used by popular services like SQL, SNTP, Apache, and others.

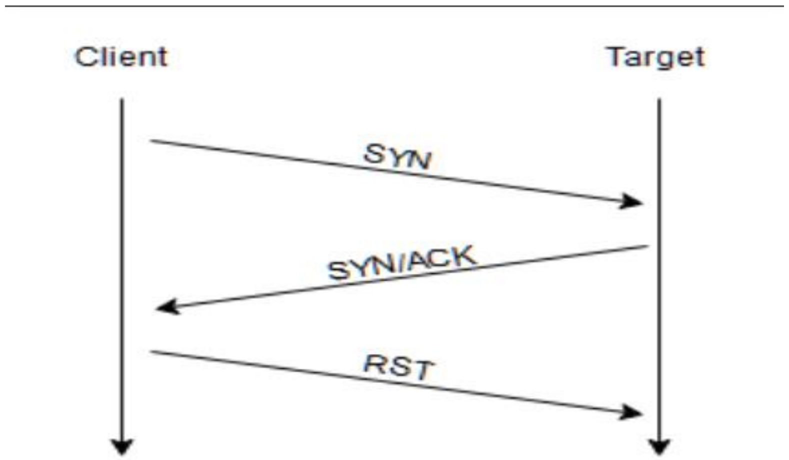
```
nmap scanme.nmap.org
```

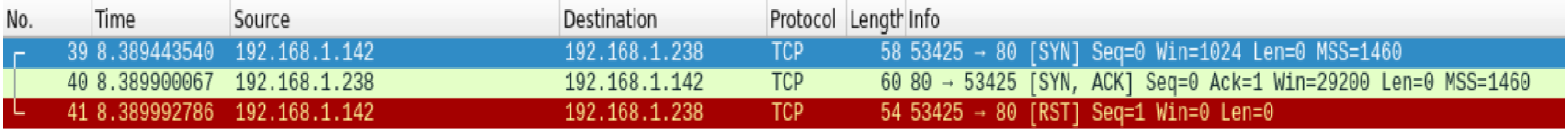
SYN scan

SYN scanning is performed by sending an SYN packet and analyzing the response. This scan is also called ‘Half-open’ scan or ‘Stealth’ scan. In this type of scan a SYN flag is send to the target port. If SYN/ACK flag is received, it means the port is open. However, a stealth scan never completes the **3-Way-Handshake**, which makes it hard for the target to determine the scanning system.

Where TCP scans performs a full three-way handshake with the target, SYN scans sends back a RST TCP packet after receiving a SYN/ACK from the server (this prevents the server from repeatedly trying to make the request).

```
# nmap -sS scanme.nmap.org
```

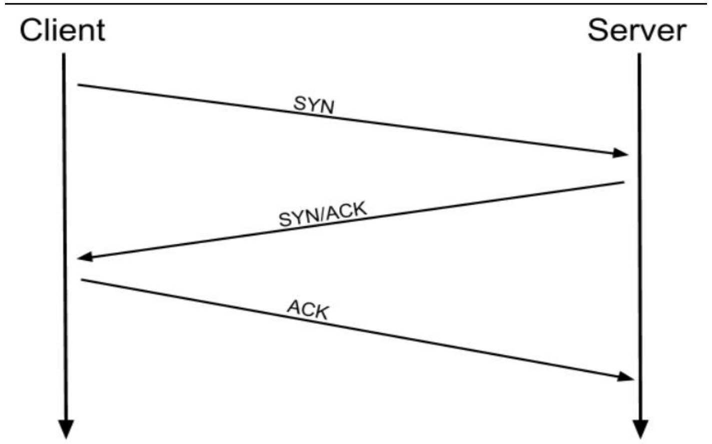


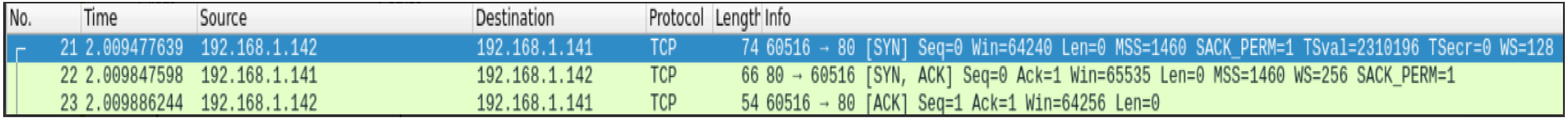


No.	Time	Source	Destination	Protocol	Length	Info
39	8.389443540	192.168.1.142	192.168.1.238	TCP	58	53425 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
40	8.389900067	192.168.1.238	192.168.1.142	TCP	60	80 → 53425 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
41	8.389992786	192.168.1.142	192.168.1.238	TCP	54	53425 → 80 [RST] Seq=1 Win=0 Len=0

TCP Connect Scan

The three-way handshake consists of three stages. First the connecting terminal (our attacking machine, in this instance) sends a TCP request to the target server with the SYN flag set. The server then acknowledges this packet with a TCP response containing the SYN flag, as well as the ACK flag. Finally, our terminal completes the handshake by sending a TCP request with the ACK flag set.





No.	Time	Source	Destination	Protocol	Length	Info
21	2.009477639	192.168.1.142	192.168.1.141	TCP	74	60516 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2310196 TSecr=0 WS=128
22	2.009847598	192.168.1.141	192.168.1.142	TCP	66	80 → 60516 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
23	2.009886244	192.168.1.142	192.168.1.141	TCP	54	60516 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0

If, however, the request is sent to an *open* port, the target will respond with a TCP packet with the SYN/ACK flags set. Nmap then marks this port as being *open* (and completes the handshake by sending back a TCP packet with ACK set).

If Nmap sends a TCP request with the *SYN* flag set to a ***closed*** port, the target server will respond with a TCP packet with the *RST* (Reset) flag set. By this response, Nmap can establish that the port is closed.

Many firewalls are configured to simply **drop** incoming packets. Nmap sends a TCP SYN request, and receives nothing back. This indicates that the port is being protected by a firewall and thus the port is considered to be *filtered*.

UDP Scan

Unlike TCP, UDP connections are *stateless*. This means that, rather than initiating a connection with a back-and-forth "handshake", UDP connections rely on sending packets to a target port and essentially hoping that they make it. This makes UDP superb for connections which rely on speed over quality (e.g. video sharing), but the lack of acknowledgement makes UDP significantly more difficult (and much slower) to scan. The switch for an Nmap UDP scan is (-sU).

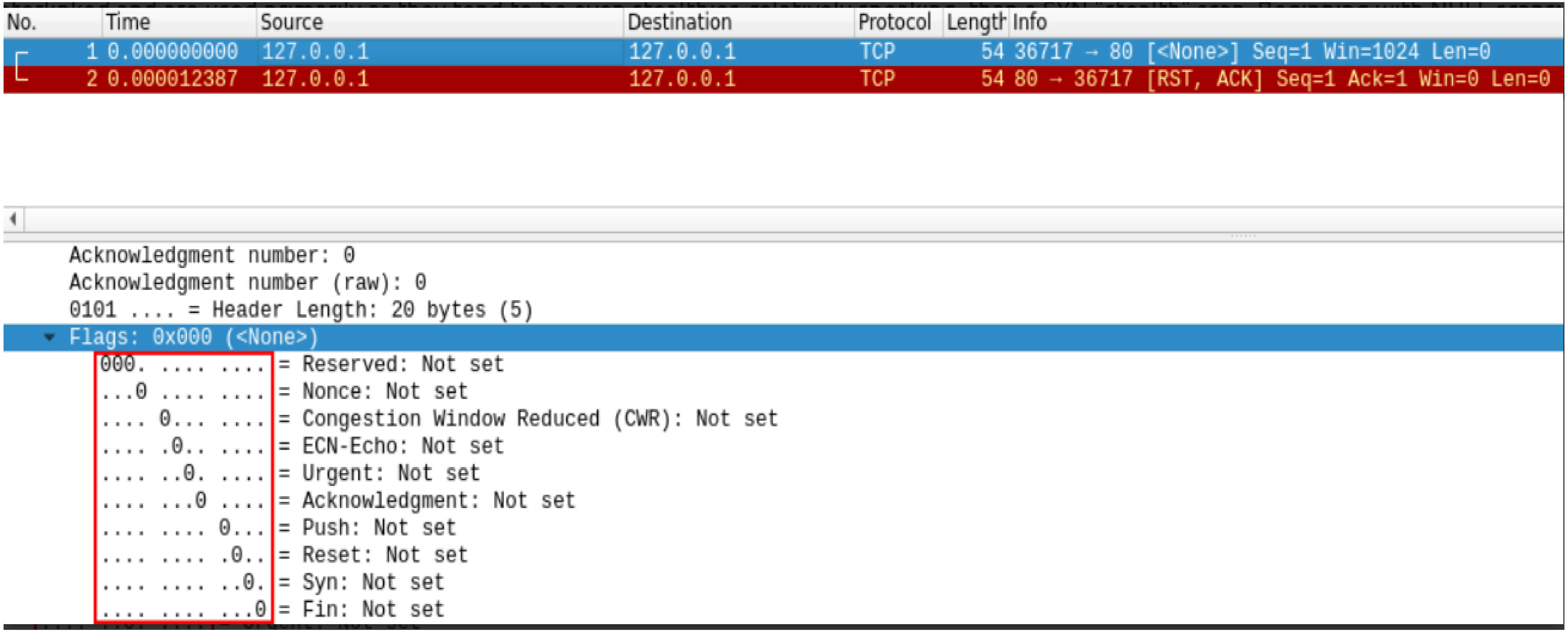
When a packet is sent to an open UDP port, there should be no response. When this happens, Nmap refers to the port as being '*open|filtered*'. In other words, it suspects that the port is open, but it could be firewalled. If it gets a UDP response (which is very unusual), then the port is marked as open. More commonly there is no response, in which case the request is sent a second time as a double-check. If there is still no response then the port is marked '*open|filtered*' and Nmap moves on.

When a packet is sent to a *closed* UDP port, the target should respond with an ICMP (ping) packet containing a message that the port is unreachable. This clearly identifies closed ports, which Nmap marks as such and moves on.

When scanning UDP ports, Nmap usually sends completely empty requests -- just raw UDP packets.

NULL Scan

As the name suggests, NULL scans (-sN) are when the TCP request is sent with no flags set at all. As per the RFC, the target host should respond with a RST if the port is closed.

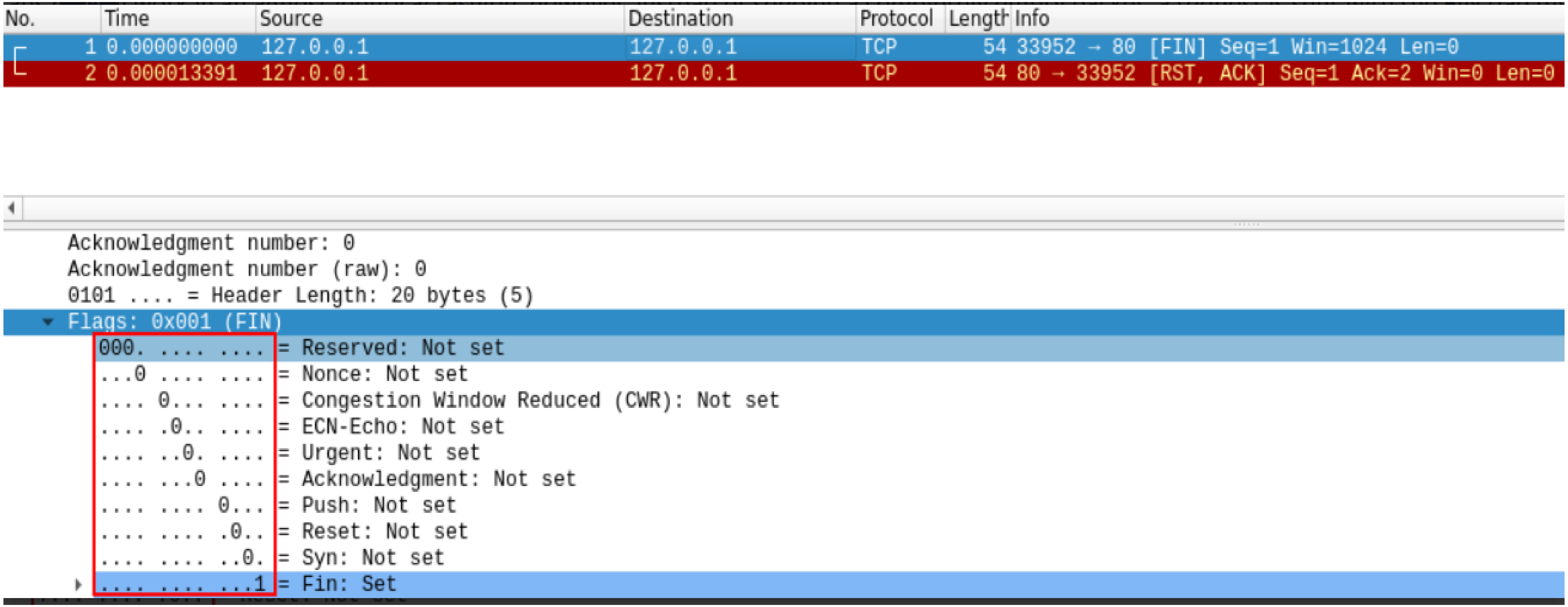


```
No.   Time          Source      Destination  Protocol Length Info
    1 0.000000000   127.0.0.1   127.0.0.1    TCP          54 36717 → 80 [<None>] Seq=1 Win=1024 Len=0
    2 0.000012387   127.0.0.1   127.0.0.1    TCP          54 80 → 36717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
```

```
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
  ▾ Flags: 0x000 (<None>)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...0 .... = Acknowledgment: Not set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
```

FIN Scan

FIN scans (-sF) work similarly as NULL scan. However, instead of sending a completely empty packet, a request is sent with the FIN flag (usually used to gracefully close an active connection). Once again, Nmap expects a RST if the port is closed.

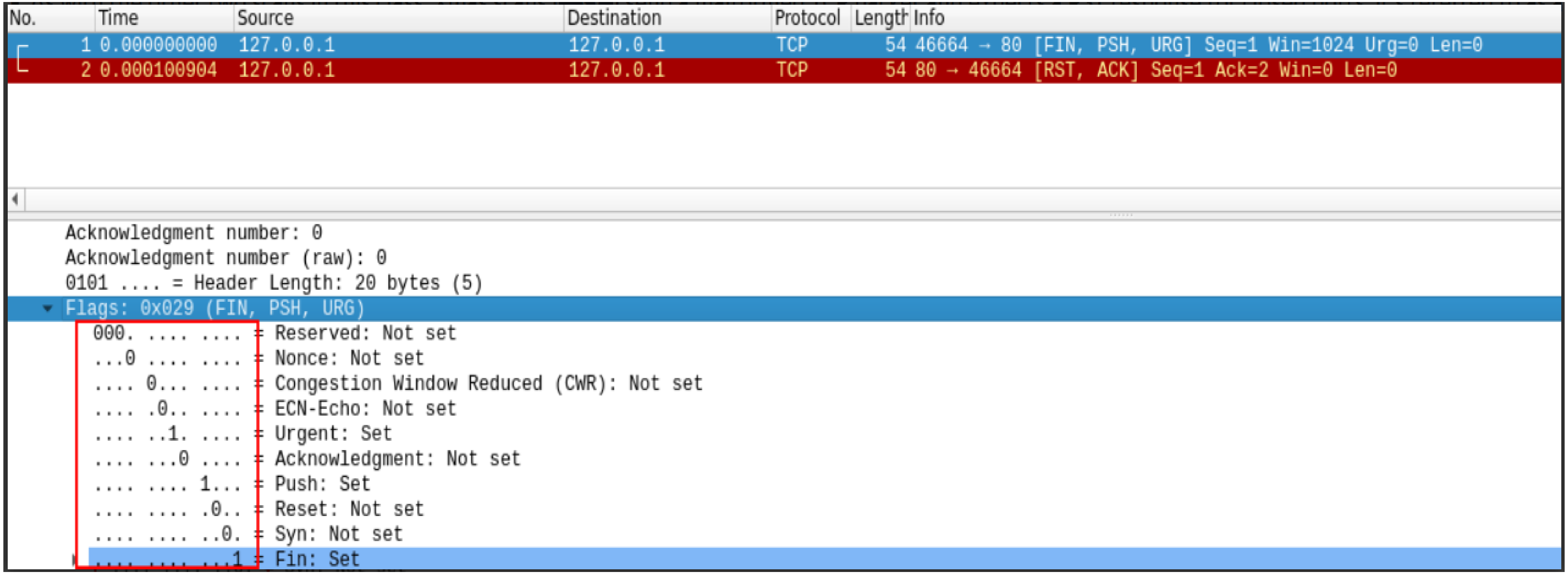


```
No.   Time         Source      Destination  Protocol Length Info
    1 0.000000000  127.0.0.1   127.0.0.1    TCP          54 33952 → 80 [FIN] Seq=1 Win=1024 Len=0
    2 0.000013391  127.0.0.1   127.0.0.1    TCP          54 80 → 33952 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
```

```
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
  ▾ Flags: 0x001 (FIN)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...0 .... = Acknowledgment: Not set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
      ▸ .... .... ...1 = Fin: Set
```

Xmas Scan

As with the other two scans (NULL, FIN), Xmas scans (-sX) send a malformed TCP packet and expects a RST response for closed ports. It's referred to as an Xmas scan as the flags that it sets (PSH, URG and FIN) give it the appearance of a blinking Christmas tree when viewed as a packet capture in Wireshark.



```
No.   Time          Source      Destination  Protocol  Length Info
    1 0.000000000   127.0.0.1   127.0.0.1    TCP       54 46664 → 80 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
    2 0.000100904   127.0.0.1   127.0.0.1    TCP       54 80 → 46664 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
```

```
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
  ▾ Flags: 0x029 (FIN, PSH, URG)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..1. .... = Urgent: Set
      .... ...0 .... = Acknowledgment: Not set
      .... .... 1... = Push: Set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...1 = Fin: Set
```

Some commonly used switches

- -sS -> SYN scan
- -sT -> TCP Connect scan
- -sU -> UDP scan
- -sF -> FIN scan
- -sN -> NULL scan
- -sX -> Xmas scan
- -sn -> No port scan (Ping scan)
- -sL -> List scan
- -Pn -> No ping
- -O -> To detect which operating system the target is running on
- -sV -> To detect the version of the services running on the target

- -A -> Enables OS detection, version detection, script scanning, and traceroute
- -v -> Increase verbosity to Level 1
- -vv or –v2 -> Increase verbosity to Level 2
- -vvv or –v3 -> Increase verbosity to Level 3
- --traceroute -> Track path to host
- -p -> Port ranges

Some commonly used switches

- --script *filename*|*category*|*directory/*|*expression* -> Accessing the scripting engine

- -T -> Set a time template
  - -T0 -> paranoid
  - -T1 -> sneaky
  - -T2 -> polite
  - -T3 -> normal
  - -T4 -> aggressive
  - -T5 -> insane

- -oA -> Use to save the nmap results in three major formats (.xml, .nmap, .gnmap)

- -oX -> Use to save the nmap results in .xml format (i.e., xml output)
  Example:- nmap -oX *myscan.xml* **target**

- -oN -> Use to save the nmap results in .nmap format (i.e., normal output)
  Example:- nmap -oN *myscan.nmap* **target**

- -oG -> Grepable output