Project 3: Network Attacks & Defenses

1. Nmap Port Scanning

Idea: Want to probe which ports are open on our target machine (victim server). Ran the nmap command below on the target machine `scanme.nmap.org`. The scan took 11 minutes over throttled stanford network. The output is shown in Figure 1.

```
sudo nmap -sS -A -T4 -p0-65535 scanme.nmap.org
```

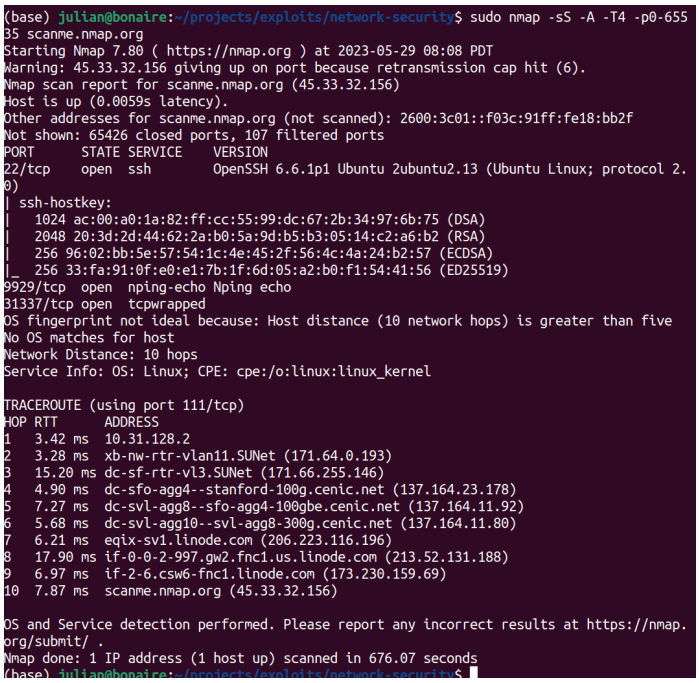

Figure 1: nmap console output

Working notes:

- Flags used: sS = TCP SYN scan; -A = OS detection (-O), version detetction (-sV), script scanning (-sC), traceroute (–traceroute); T4 = quick scan; -p0-65535 = expand scan from top 1000 to all ports
- Relevant websites: `https://nmap.org/book/man-version-detection.html` and `https://nmap.org/book/man-port-scanning-techniques.html`

2. Wireshark Packet Sniffing

Idea: Wireshark is a tool to monitor and record local network traffic (packet sniffing). We start recording packets, then run the nmap command from part 1 and stop recording

after nmap completes its scan. Export the wireshark output as a .pcap file and open it in wireshark to analyze.

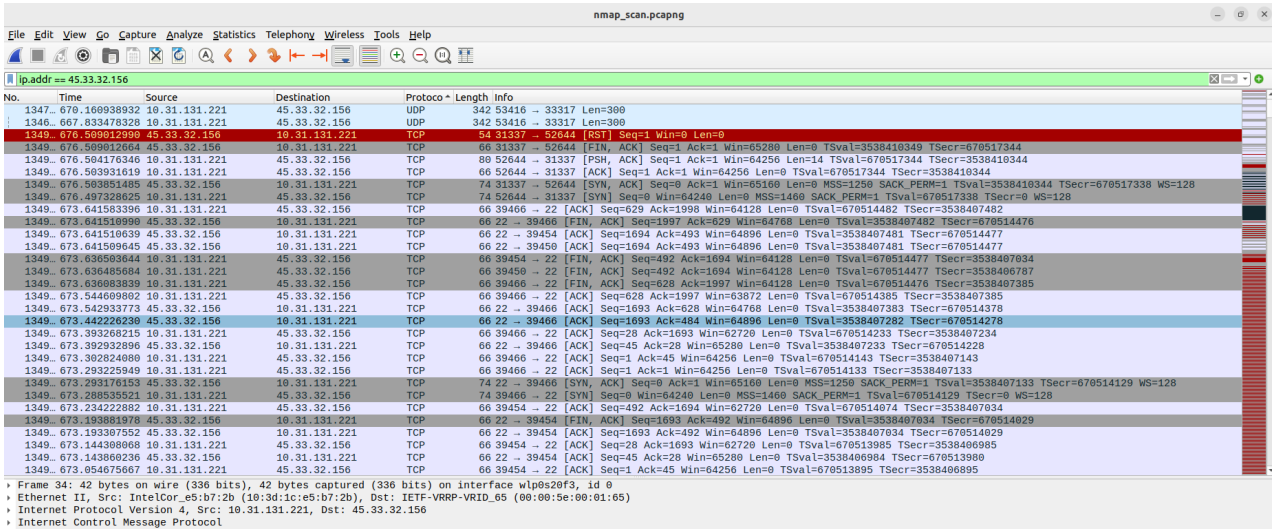

Figure 2: wireshark gui output

Working notes:

- A port is ‘closed‘ if no application or service from target machine is listening on that port. We can identify closed ports on wireshark when the target machine responds immediately with an RST,ACK packet, or NIL if not packet is sent.
- A port is ‘filtered‘ if your probe did not reach the destination port because it was dropped by the firewall. We can identify filtered ports on wireshark when the target machine responds with no response packet. This is sometimes followed by an ICMP communication to confirm that the port is unreachable.
- To find which types of http requests were made by nmap, we simply filter on HTTP for protocol field. There are HTTP GET, HTTP OPTIONS, HTTP POST, and HTTP PROPFIND requests made during the scan.
- To check TCP parameters changed by nmap, scroll through wireshark output and inspect TCP packets. We find that only Destination Port, [Stream Index], Sequence Number (raw), Checksum are changed.

3. Programmatic Packet Processing

Idea: Programmatically analyze a PCAP (Packet Capture) file to detect Port Scanning and ARP Spoofing. Port Scanning is when an attacker uses `nmap` (or similar) to find all open ports on a known host. ARP Spoofing is when an attacker sends ARP packets to a target machine to redirect traffic to a different machine.

```
code goes here
```

Working notes:

- xx
- xx