Analyzing Reconnaissance Threats to Computing and Network Environments Labs

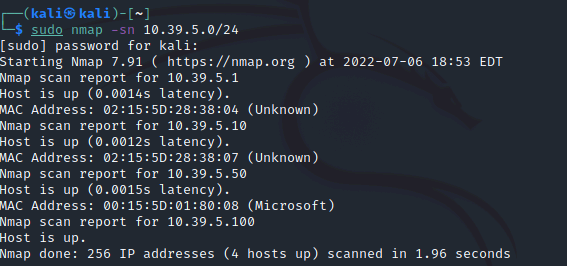
LAB-1

Scenario: You want to see how attackers can execute a reconnaissance attack. You'll scan your network and hosts to see the kind of useful intelligence an attacker can glean. Understanding the nature of these reconnaissance threats will enable you and your team to eliminate weaknesses in your network that reveal too much information.

Environment being used: Kali Linux 10.39.5.100, Windows 10 10.39.5.10 and Windows Server 2019 VM 10.39.5.50

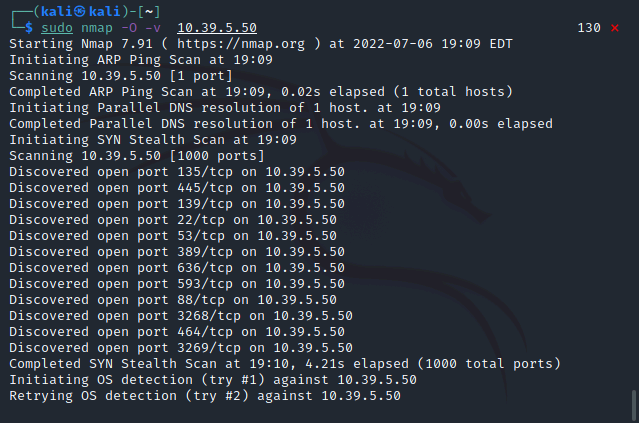
Lab1: Using Kali Linux to operate Nmap through terminal to collect information about other systems in the space such as Windows 10/ Server 2019

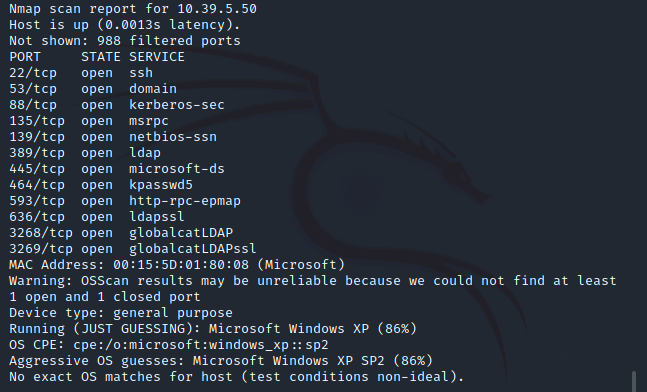
1. Logged into Kali Linux VM to utilizing Terminal to utilized NMAP.
   1. To view Nmap’s properties and descriptions, issue just the Nmap command or for the man pages issue the man Nmap command.
      1. Syntax: [Scan Type] [Option] [Target Specified]
   2. Question: Looking under host discovery which options can be used to issue a simple ping scan? 🡪 Nmap -Sn
   3. Question: Looking under Scan Techniques what options runs a TCP Connect () Scan? 🡪 Nmap -sT
   4. Question: Looking under OS Detection, what option can be run an OS discovery scan? 🡪 Nmap -O
   5. Question: looking under OUTPUT what does -v option mean? 🡪 Option -v provides more/additional information than being displayed
2. Issuing ping scan against local network



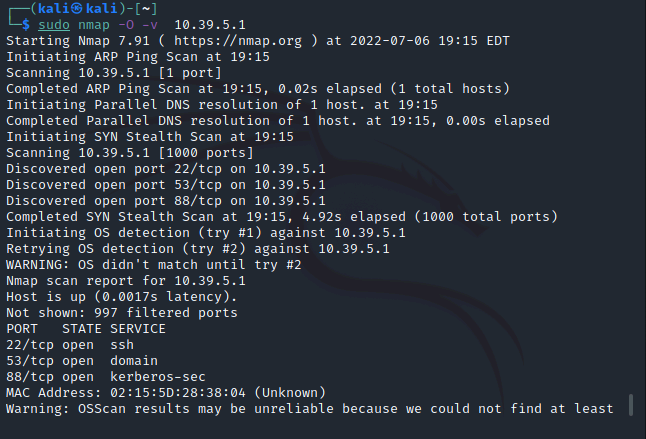
Note: When using Sudo, the system will prompt you for the systems password every 5 minutes

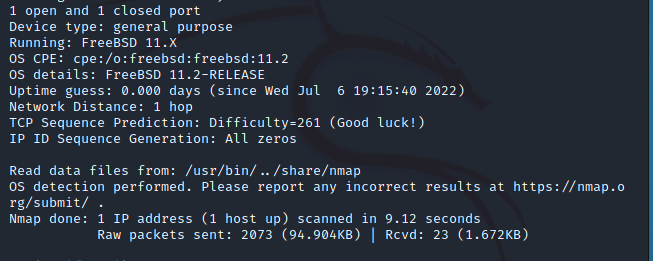
1. Issuing discovery scan to determine systems Operating System using the Sudo Nmap -O -v 10.39.5.50 command.



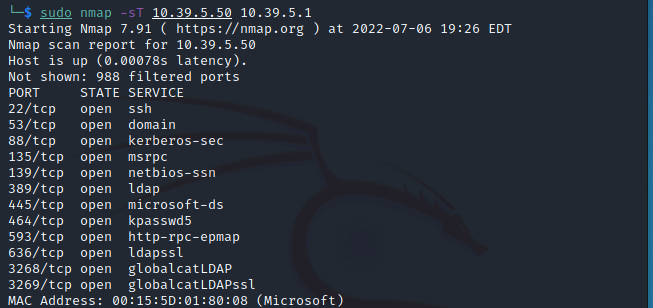


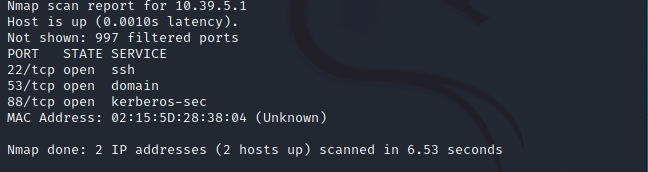
1. Issuing discovery scan to determine systems Networking devices- Router





1. Ran a TCP () Connect scan against both server and router. Which host shows more ports and why
   1. Note: You can include multiple IPs to search in one line by simply putting a space in between each address



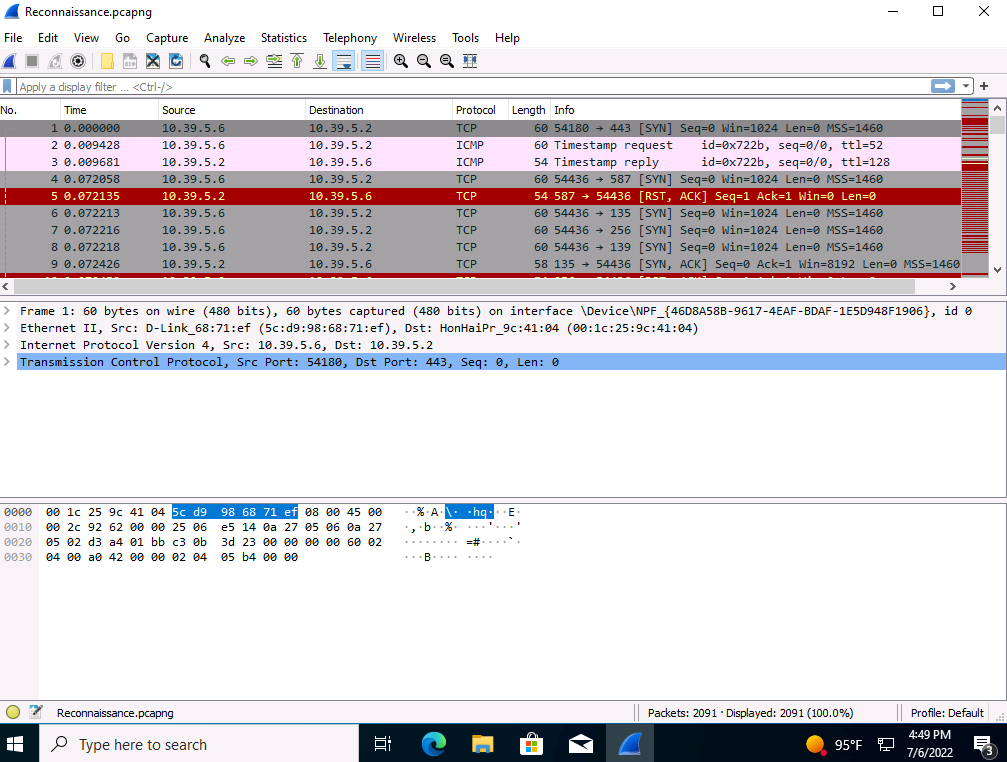


Answer to Question: The server displayed more port numbers because it is a general-purpose system where as the networking device router is a dedicated device

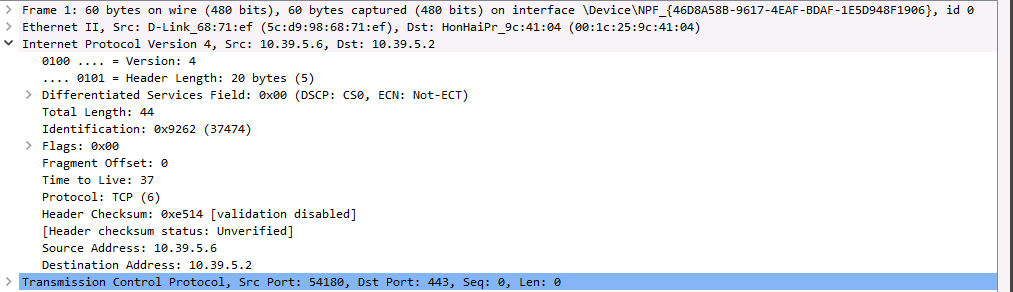
LAB 2:

Scenario: Using Wireshark on Windows 10 VM Client to analyze previously captured packets

1. Logged into Client VM and started Wireshark for analysis
2. Opened saved Capture.

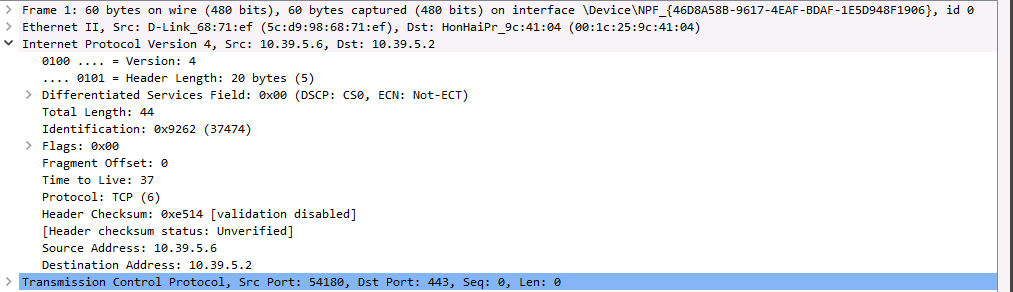


1. Looking at the first packet in the list, what is the source and destination IP address



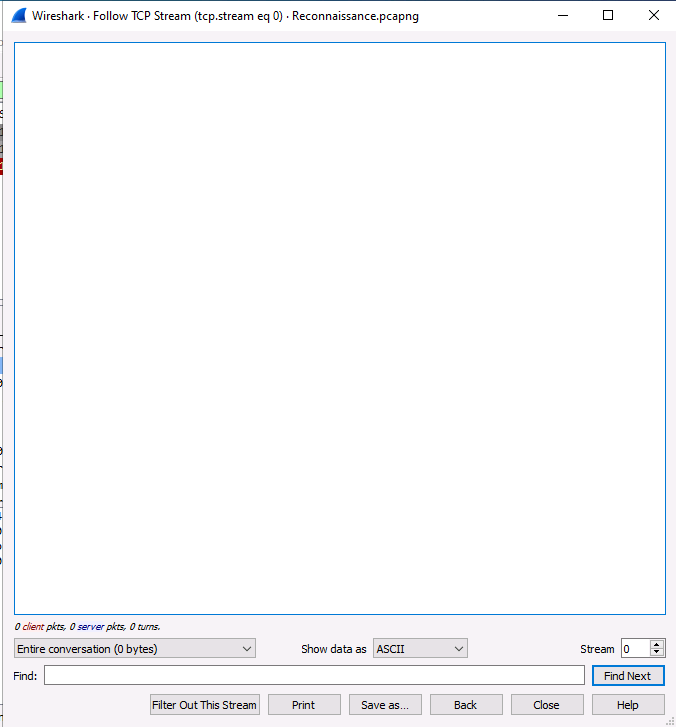
Source IP: 10.39.5.6 Destination IP: 10.39.5.2

1. Looking at the TCP section, what is the destination port? Flags sent in the packet ?



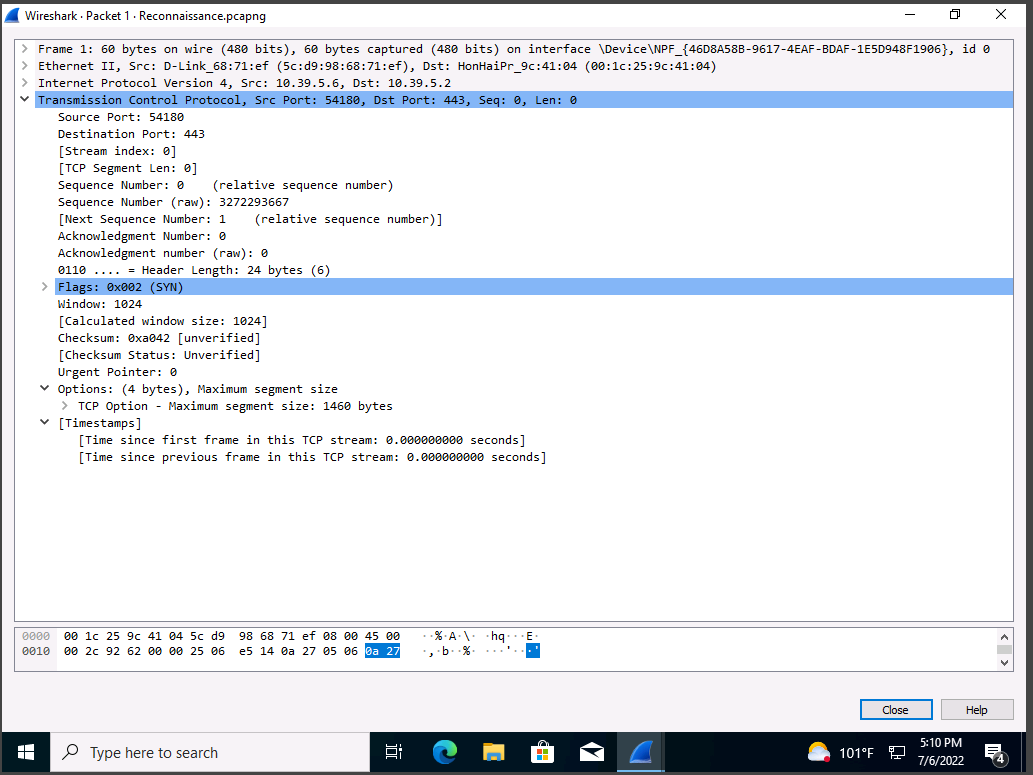
Destination Port: 443 (HTTP) Flags: 0x002 (SYN) which is the beginning to establish a connection with a host or service

1. Analyzed the capture for attacks.
   1. Went to Statistics 🡪 Conversations
   2. TCP tab 🡪 Sort Packets looking to see how many packets came from each IP. Then Sorting Port B (Destination Ports) from descending order
   3. Made note of the Various destination port numbers as I continued my analysis
   4. Checking the TCP Stream, but in this case, there is no information

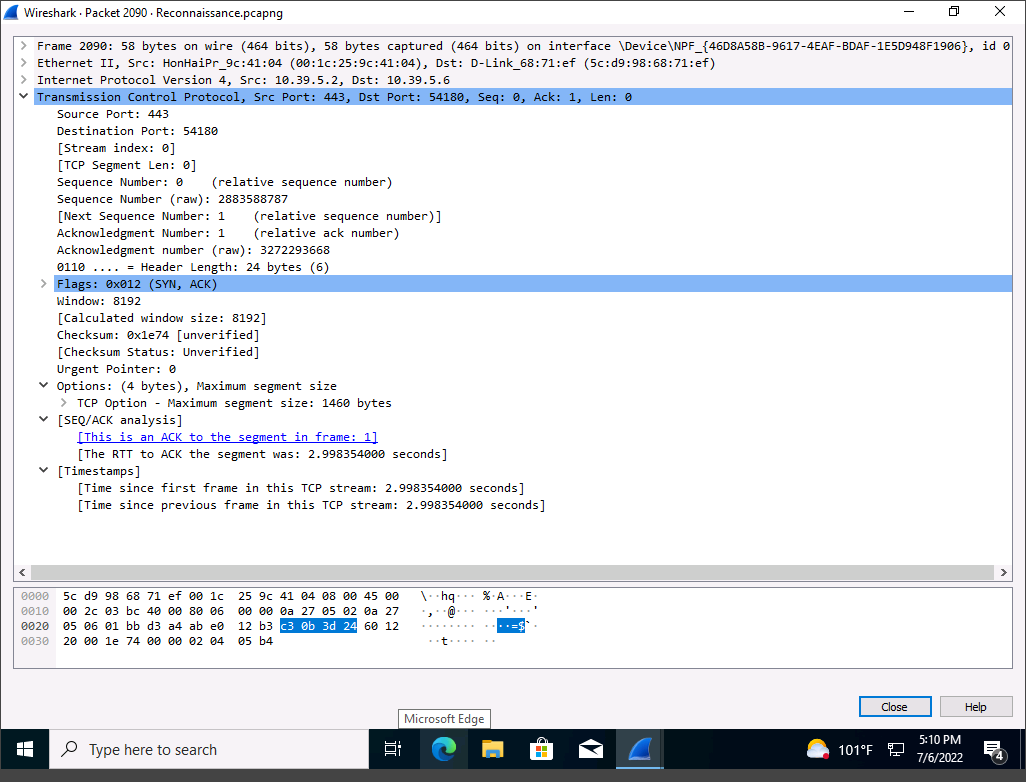


The Stream did not generate anything because there was no communication in that TCP session, yet it did filter the packet to more of a manageable number of packets to analyze.

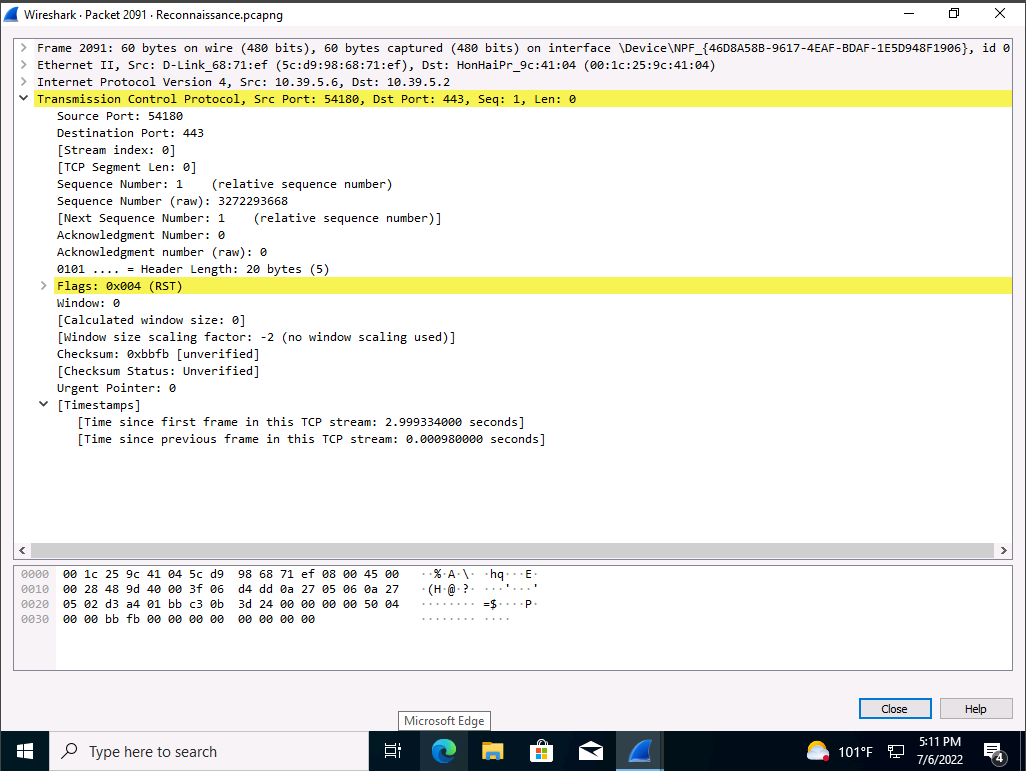
Packet-1



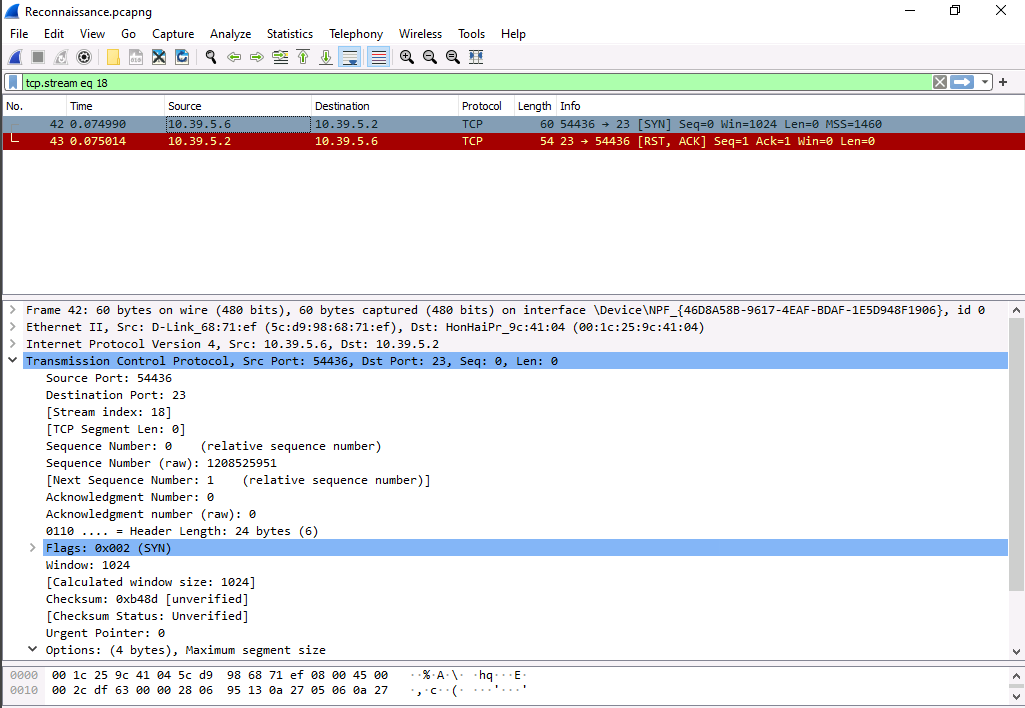
Packet-2



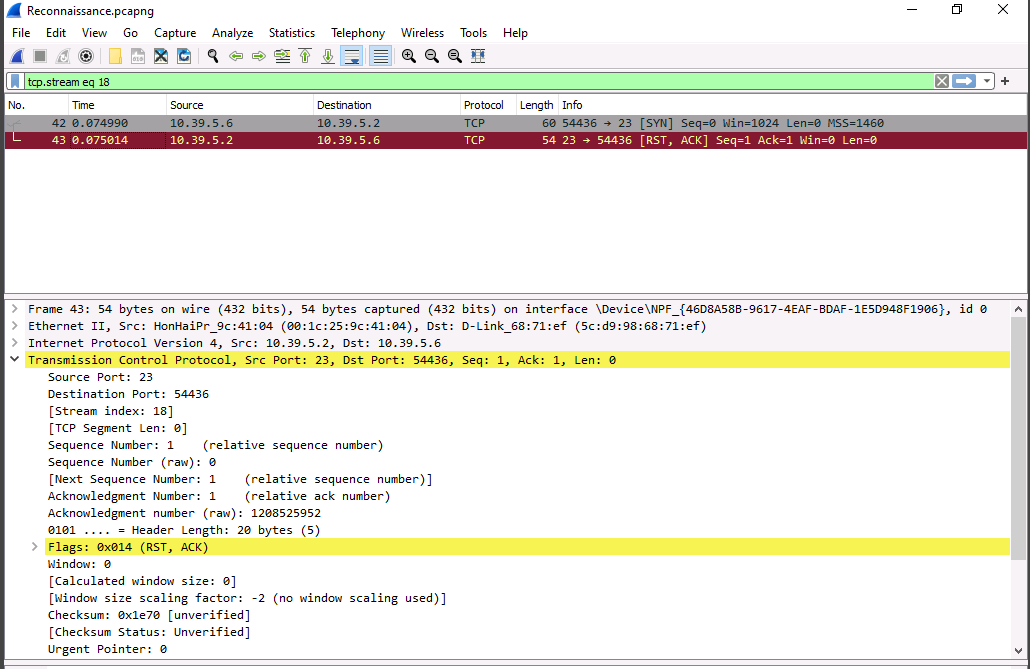
Packet-3



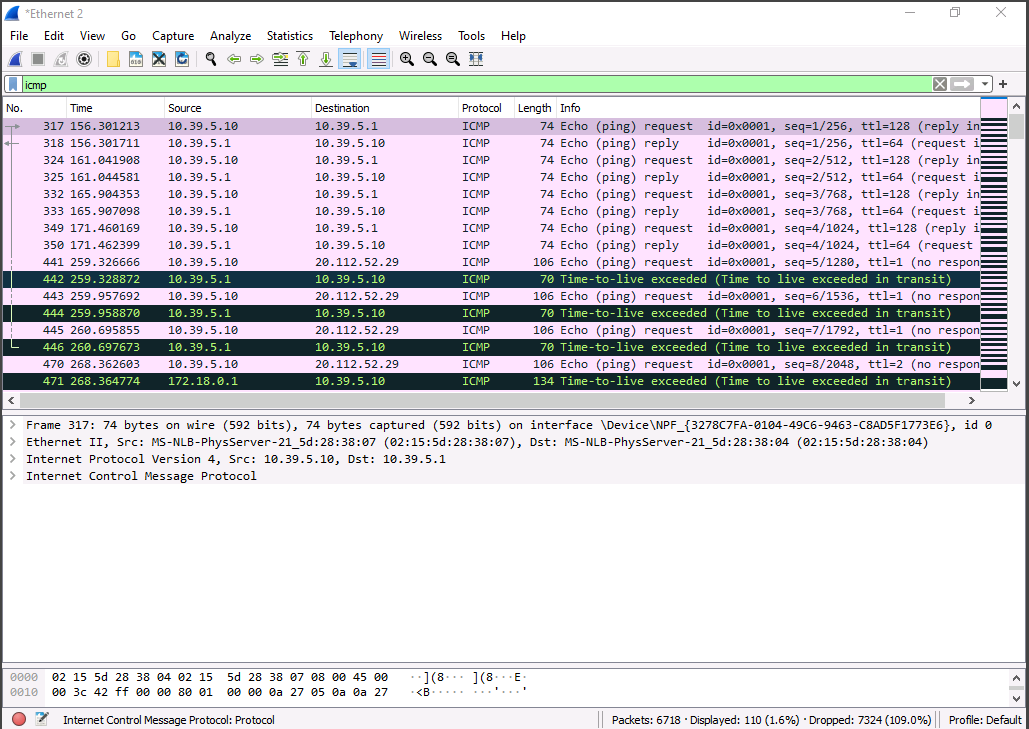
From analyzing the three different packets I have concluded that the attacker initiated a session (SYN) to see if they could get a response from the server in which packet 2 verifies that the server in fact responded to the SYN request with its own (SYN+ACK). From there we can see that the attack got what they were looking for as analyzing the third packet we see that they issue a Restart (RST) Response resulting in reconnaissance attack. Since it interrupted before the session could be completed, we can say that it was Stealth Scan



This task was getting more exposure to filtering network traffic in the logs for inspection. I was searching for TCP traffic and this happened to be the first packet after being filtered. From inspecting the log information and looking at packet information I’m able to determine this was a packet from an attacker that was attempting to start a conversation over port 23, Telnet.

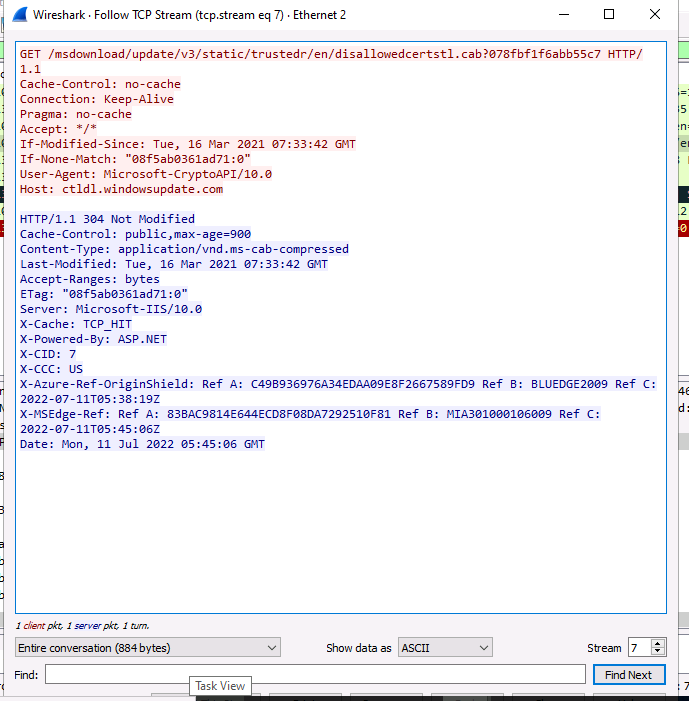


Analyzing the next packet in connection to the one above I see that the server was blocked with a reset response. This also confirms that the security policies and devices are configured correctly and doing their respective job.



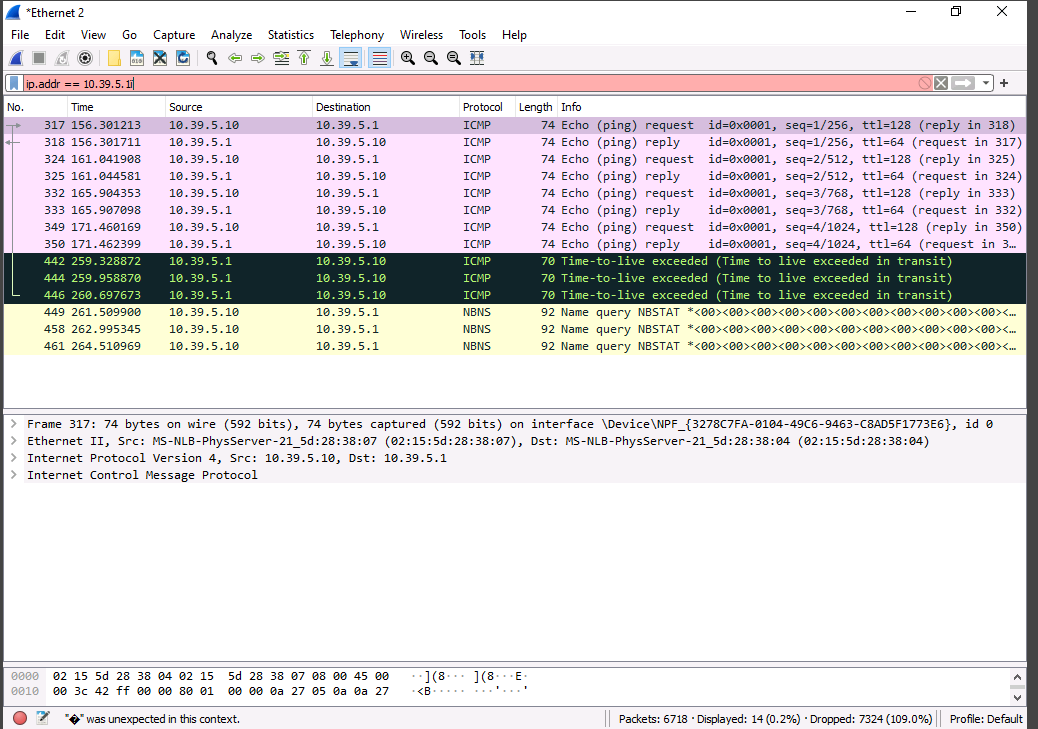
In this portion of the lab I filtered traffic to be analyzed. I started out by running a scan to collect the results. I kept the scan going but went into Windows PowerShell to ping the server and run a traceroute to windows.com to generate some web traffic for inspection and analysis. I filtered the searched by ICMP based on Ping and Traceroute both operate using ICMP packets

Looking at the results from the scan I see the routes being displayed that resulted from the traceroute command being issued. I am seeing some errors depicted in black, but its too early too tell if those are actual errors due to the face its coming from our traceroute which can be said it to be from a delay in discovering a path.

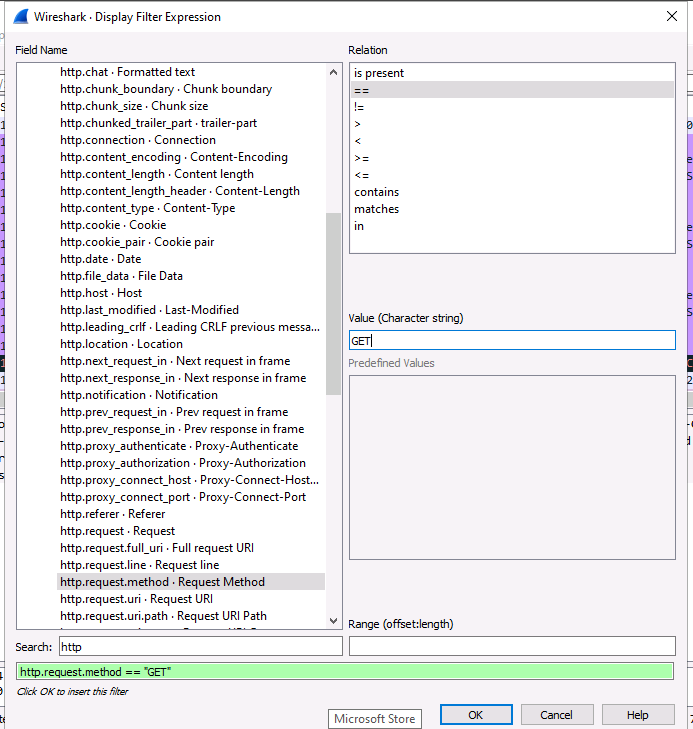


Inspecting the packet results I’m seeing a lot of web traffic, which tells me I need to inspect HTTP traffic. I then filtered to show only http traffic using “http” command. Off bat didn’t see any anomalies so I decided to look at the TCP stream. This was achieved by selecting any http packet, right clicking, selecting follow and TCP stream.

Note: Red = Client-Side | Blue = Server-Side.



Here I am exploring with filtering using an IP.



To get to this screen in Wireshark go to Analyze 🡪 Display Filter Expression. This method assists in creating useful search filters. This acts like a man, help or a? page where you are presented with a list then you just search and input the necessary data

Note: You can combine filters together using || Pipe for or, && for and.

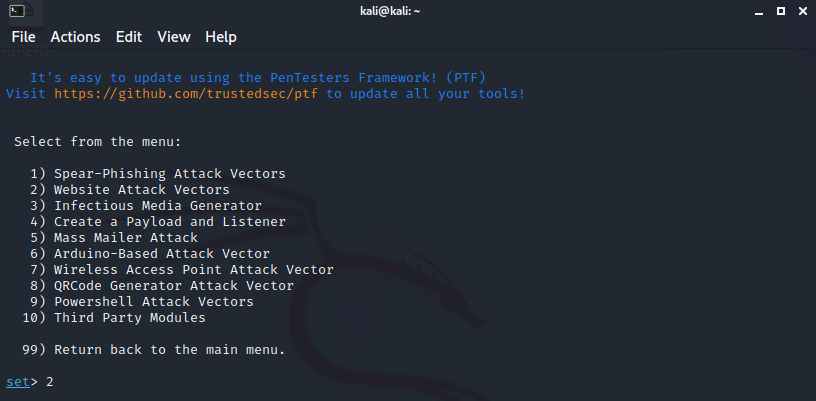
Lab 5: Assessing the impact of social engineering.

Scenario: Several employees recently had some of their personal credentials stolen. These credentials were to major sites like Google, Facebook, and LinkedIn. All the victims claim that, in accordance with company security policy, they never directly gave their user names and passwords to anyone asking for them. You therefore suspect they were tricked in a more subtle way—that the websites they thought they were logging in to were in fact convincing forgeries.

In order to assess how effective pharming attacks are on your personnel, you'll see just how easy it is to spoof the sign-in page of a major public website. For now, it was just the employees' personal accounts that were compromised—but you don't want this to happen when they log in to an internal website with their work credentials.

Requirements: Kali Linux Machine, Social-Engineer Toolkit, Python-based exploit framework.

To begin with I spun up the Kali Linux machine I was using. Once booted up I had opened a terminal to issue sudo setoolkit (This is for the Social-Engineering toolkit) command.

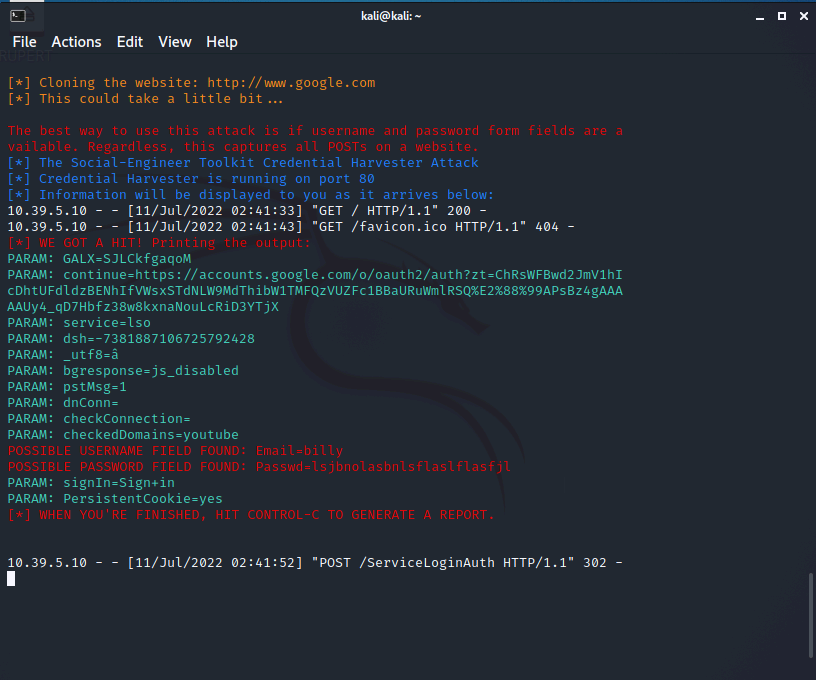


Once the exploit is loaded, its self-explanatory in following the screen menu instructions. For example, I am wanting to see social engineering attacks. At the main page I selected 1 for social engineering attacks, 2 for web attacks, 3 credential harvester attack to gather the creds from the victim.



Once I was done configuring the exploit, I had to then configure the fake/Malicious server. Therefore, I selected the web template then at prompt I inputted my server IP.

The final step is to trick a user. To demonstrate this activity, I logged back into my Windows 10 VM brought up a browser to go to the malicious IP to see If I configured it correctly. After seeing that the lab was progressing as attended, I entered random test credentials.



Lastly, it was to verify that the exploit worked. Reviewing the output, I do see credentials from username entered was Billy and the password inputted was a random string, but still vailed