Lab 5b: PCAP Analysis with Wireshark

Grade: 7% (100 Points)

***Note**: This assignment requires research and independent learning. You are encouraged to use web resources, official documentation, and tutorials to understand the functionality of each tool and protocols. The goal is to analyze the packet and find information about vulnerability.*

What is PCAP?

PCAP stands for Packet Capture, which is a file format used to store network packet data captured from a network interface. It is commonly associated with network analysis and troubleshooting activities.

PCAP files contain the raw data of network packets, including the headers and payloads of each packet. These files can be generated by packet capture tools such as Wireshark, tcpdump, or other network monitoring software.

PCAP files are widely used in network analysis and security tasks. They enable network administrators, analysts, and researchers to inspect and analyze network traffic for various purposes, including:

1. Network troubleshooting: PCAP files can help diagnose network issues by examining packet-level details such as source and destination addresses, protocols, and error messages.
2. Network security: PCAP files are valuable for detecting and investigating network security incidents. They allow security professionals to analyze packet payloads, identify malicious activity, and track network intrusions.
3. Protocol analysis: PCAP files provide a wealth of information about network protocols. By analyzing the captured packets, researchers can gain insights into the behavior of network protocols, identify vulnerabilities, and develop mitigation strategies.
4. Performance monitoring: PCAP files can be used to measure network performance, identify bottlenecks, and optimize network configurations. They provide a detailed view of network traffic, allowing administrators to analyze latency, throughput, and other performance metrics.

To capture PCAP files you need to use a packet sniffer. A packet sniffer captures packets and presents them in a way that's easy to understand. When using a PCAP sniffer the first thing you need to do is identify what interface you want to sniff on.

Using Wireshark for PCAP file capture and analysis

Wireshark is the most popular traffic analyzer in the world. Wireshark uses .pcap files to record packet data that has been pulled from a network scan. Packet data is recorded In files with the .pcap file extension and can be used to find performance problems and cyberattacks on the network.

In other words, the PCAP file creates a record of network data that you can view through Wireshark. You can then assess the status of the network and identify if there are any service issues that you need to respond to.

It is important to note that Wireshark isn't the only tool that can open .pcap files. Other widely used alternatives include tcpdump and WinDump, network monitoring tools that also use PCAP to take a magnifying glass to network performance.

Download the PCAP file provided and use your analysis tools to examine the provided PCAP file.

Scenario:

You, as a SOC analyst, belong to a company specializing in hosting web applications through KVM-based Virtual Machines. Over the weekend, one VM went down, and the site administrators fear this might be the result of malicious activity. They extracted a few logs from the environment in hopes that you might be able to determine what happened.

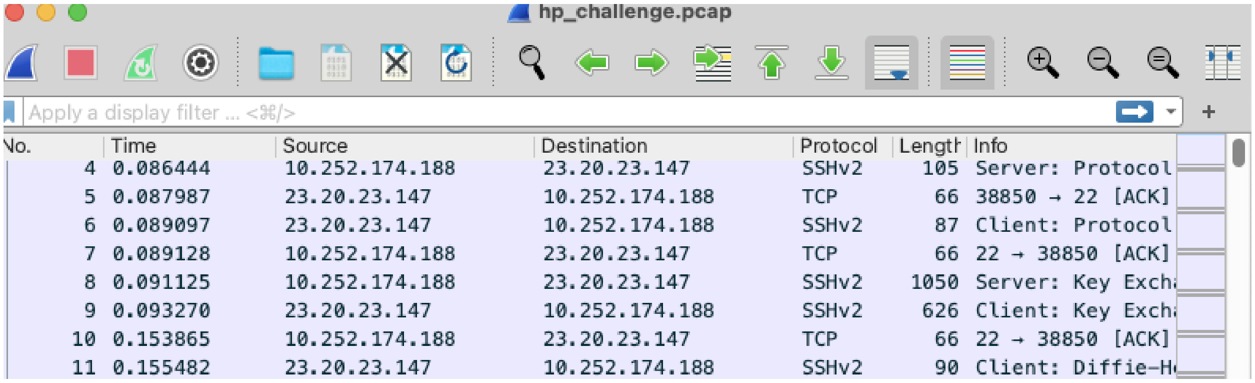
This challenge is a combination of several entry to intermediate-level tasks of increasing difficulty focusing on authentication, information hiding, and cryptography. Participants will benefit from entry-level knowledge in these fields, as well as knowledge of general Linux operations, kernel modules, a scripting language, and reverse engineering. Not everything may be as it seems. Innocuous files may turn out to be malicious so take precautions when dealing with any files from this challenge. In this lab, you will analyze pcap files from the scenario, whereas in the upcoming labs, you will analyze the log files.

Download the pcap file and use your analysis tools to examine provided PCAPs and log files.

Challenge Questions

Open hp_challen.pcap in Wireshark provided by the CTF challenge.

Next, use Wireshark to open the PCAP file and see if there is **SSH** traffic in the network capture file.



```
No.  Time      Source          Destination     Protocol  Length Info
  4  0.086444  10.252.174.188  23.20.23.147    SSHv2       105  Server: Protocol
  5  0.087987  23.20.23.147    10.252.174.188  TCP          66  38850 → 22 [ACK]
  6  0.089097  23.20.23.147    10.252.174.188  SSHv2        87  Client: Protocol
  7  0.089128  10.252.174.188  23.20.23.147    TCP          66  22 → 38850 [ACK]
  8  0.091125  10.252.174.188  23.20.23.147    SSHv2      1050  Server: Key Exch
  9  0.093270  23.20.23.147    10.252.174.188  SSHv2       626  Client: Key Exch
 10  0.153865  10.252.174.188  23.20.23.147    TCP          66  22 → 38850 [ACK]
 11  0.155482  23.20.23.147    10.252.174.188  SSHv2        90  Client: Diffie-H
```

1. (5 Points) What is SSH Protocol? What is the function of this protocol?

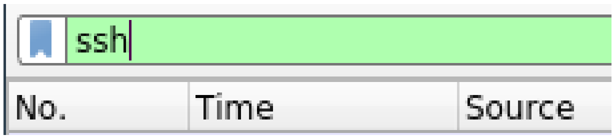
SSH Stands for secure shell which provides secure, encrypted transmission and communication between two computers over an unsecured network. It is often used for remote connection and command execution.

2. (5 Points) What other protocols can you see? List all of them.

SSHv2, TCP, HTTP

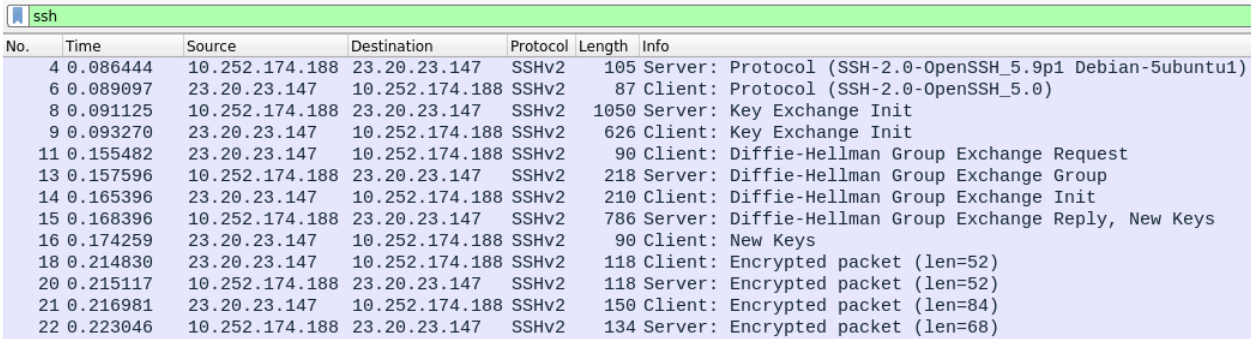
3. (10 Points) Find out what type of attack was used to gain access to the system? Explain.
   Hint: filter out all the ssh packet

Based on the timing of the ssh requests and the exchange of keys, whoever was sending the data was authorized so the likely reason for the type of attack would be a credential theft attack from either a brute force attack or stolen credentials of the username/ password. According to google, the attacker could have exploited a vulnerability in OpenSSH 5.9p1 (CVE-2014-2532) to bypass authentication. This would allow them to establish the SSH session without needing valid credentials



```
ssh
No.        Time         Source
```

You can see that the initial output from this filter shows multiple failed attempts to establish SSH sessions.



```
ssh
No.  Time       Source          Destination     Protocol Length Info
   4 0.086444   10.252.174.188  23.20.23.147    SSHv2     105 Server: Protocol (SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1)
   6 0.089097   23.20.23.147    10.252.174.188  SSHv2      87 Client: Protocol (SSH-2.0-OpenSSH_5.0)
   8 0.091125   10.252.174.188  23.20.23.147    SSHv2    1050 Server: Key Exchange Init
   9 0.093270   23.20.23.147    10.252.174.188  SSHv2     626 Client: Key Exchange Init
  11 0.155482   23.20.23.147    10.252.174.188  SSHv2      90 Client: Diffie-Hellman Group Exchange Request
  13 0.157596   10.252.174.188  23.20.23.147    SSHv2     218 Server: Diffie-Hellman Group Exchange Group
  14 0.165396   23.20.23.147    10.252.174.188  SSHv2     210 Client: Diffie-Hellman Group Exchange Init
  15 0.168396   10.252.174.188  23.20.23.147    SSHv2     786 Server: Diffie-Hellman Group Exchange Reply, New Keys
  16 0.174259   23.20.23.147    10.252.174.188  SSHv2      90 Client: New Keys
  18 0.214830   23.20.23.147    10.252.174.188  SSHv2     118 Client: Encrypted packet (len=52)
  20 0.215117   10.252.174.188  23.20.23.147    SSHv2     118 Server: Encrypted packet (len=52)
  21 0.216981   23.20.23.147    10.252.174.188  SSHv2     150 Client: Encrypted packet (len=84)
  22 0.223046   10.252.174.188  23.20.23.147    SSHv2     134 Server: Encrypted packet (len=68)
```

The image above represents the different steps that take place when attempting to establish and SSH session. These steps are briefly outlined below:

a. The client and server **negotiate the SSH version** (*i.e. packet no. 4 & 6*).
b. The client and server **exchanged public keys to generate secret key**. The server then issues a "*New Keys*" message and waits for the client to answer. (*i.e. packet no. 8, 9, 11, 13, 14 and 15*).
c. The client **acknowledges the server's "*New Keys*" message** (*i.e. packet no. 16*)
d. We then see several **encrypted packets** before the SSH session is closed (*i.e. packet no. 18, 20, 21 and 22*).

Looking down through the SSH traffic, we see this process repeated multiple times until we near the end of the SSH filtered output. At packet number **1365**, we see an attempt to establish an SSH session, only this time we see far more encrypted packets then with previous attempts. If you look at the lower packets it shows guessing attacks. In cryptography, what is the attack that consists of an attacker submitting many passwords or passphrases with the hope of eventually guessing correctly.

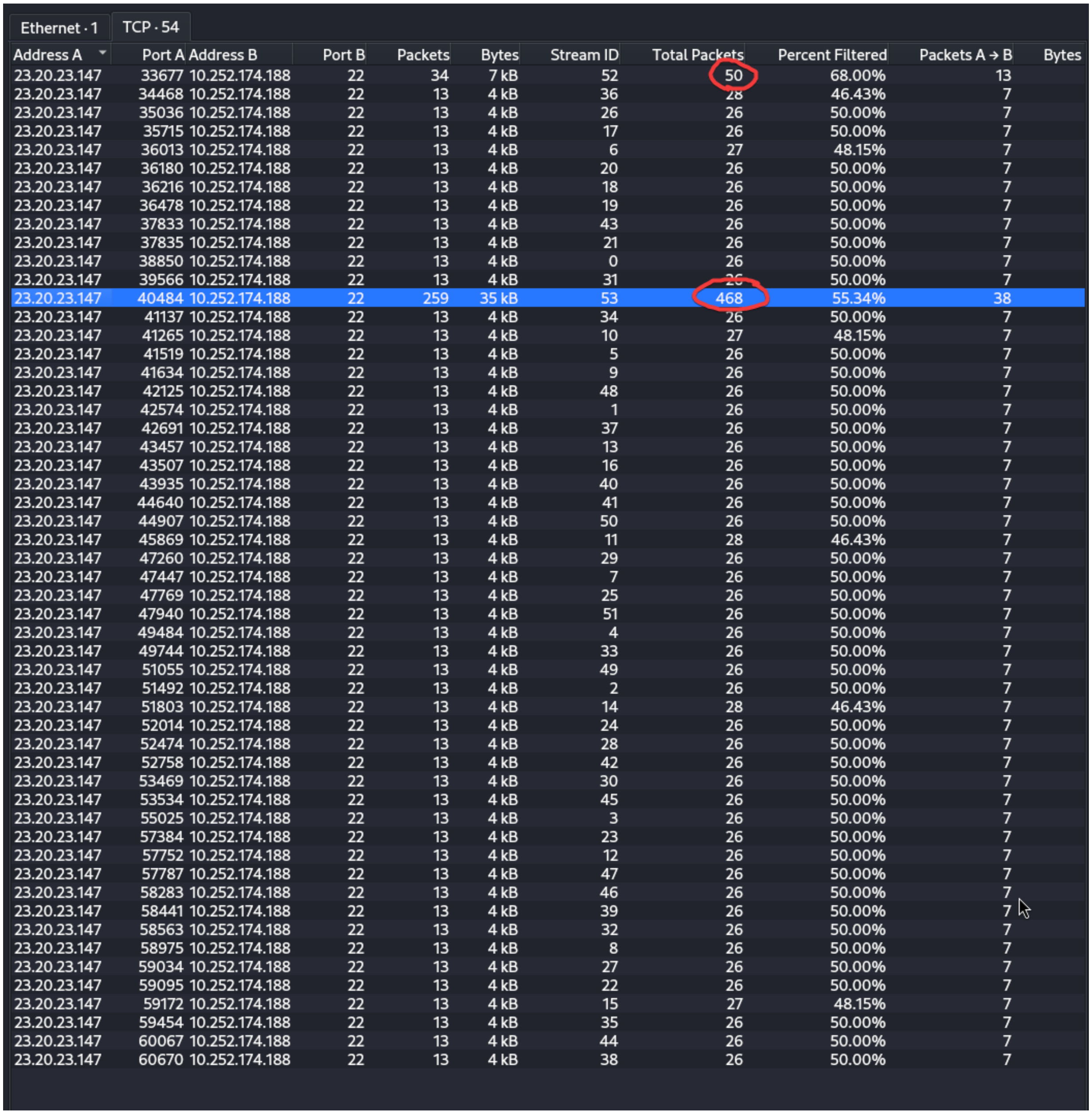
Read more : https://resources.infosecinstitute.com/topics/incident-response-resources/network-traffic-analysis-for-ir-ssh-protocol-with-wireshark/?source=post_page-----ea7abcc68a18--------------------------------

4. (10 Points) What was the tool the attacker possibly used to perform this attack?
   Hint: common linux bruteforce tool, Google it!

The common linux brute forcing tool which could be used for this attack would be the hydra ssh brute force or dictionary attack.

5. (10 Points) Now, find out how many failed attempts were there.

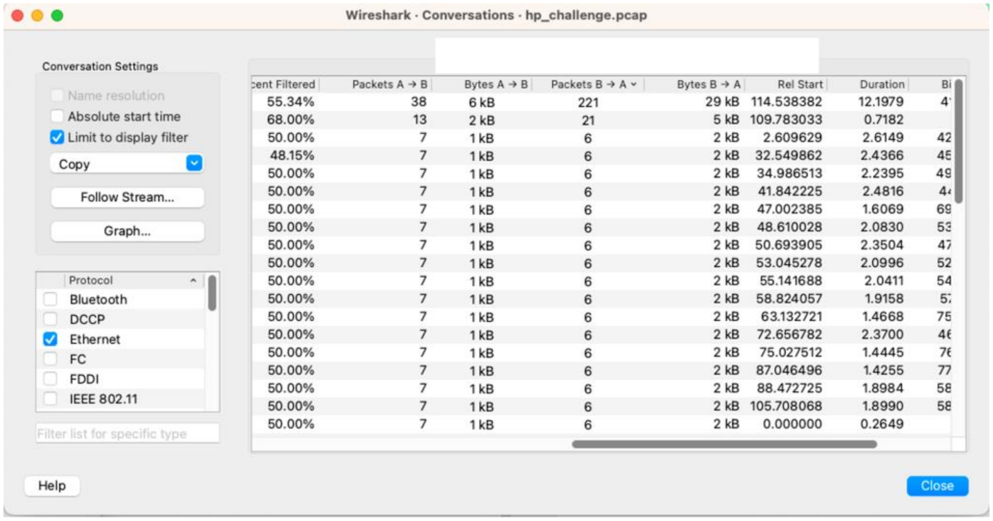
In the top left of my wireshark it says TCP * 54 meaning 54 total attempts were made. Looking at the photo, we know that the first row was a success since it's more packets than the rest since it completes the handshake, negotiates encryption, and exchanges encrypted data while the other rows don't have a large amount of data. We also see the highlighted blue row to have 468 packets sent which indicates a brute force attack over ssh probably using hyrda. Since there are 54 rows, 2 successful, there are 52 failed attempts.



Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Total Packets	Percent Filtered	Packets A → B	Bytes
23.20.23.147	33677	10.252.174.188	22	34	7 kB	52	50	68.00%	13	
23.20.23.147	34468	10.252.174.188	22	13	4 kB	36	28	46.43%	7	
23.20.23.147	35036	10.252.174.188	22	13	4 kB	26	26	50.00%	7	
23.20.23.147	35715	10.252.174.188	22	13	4 kB	17	26	50.00%	7	
23.20.23.147	36013	10.252.174.188	22	13	4 kB	6	27	48.15%	7	
23.20.23.147	36180	10.252.174.188	22	13	4 kB	20	26	50.00%	7	
23.20.23.147	36216	10.252.174.188	22	13	4 kB	18	26	50.00%	7	
23.20.23.147	36478	10.252.174.188	22	13	4 kB	19	26	50.00%	7	
23.20.23.147	37833	10.252.174.188	22	13	4 kB	43	26	50.00%	7	
23.20.23.147	37835	10.252.174.188	22	13	4 kB	21	26	50.00%	7	
23.20.23.147	38850	10.252.174.188	22	13	4 kB	0	26	50.00%	7	
23.20.23.147	39566	10.252.174.188	22	13	4 kB	31	26	50.00%	7	
23.20.23.147	40484	10.252.174.188	22	259	35 kB	53	468	55.34%	38	
23.20.23.147	41137	10.252.174.188	22	13	4 kB	34	26	50.00%	7	
23.20.23.147	41265	10.252.174.188	22	13	4 kB	10	27	48.15%	7	
23.20.23.147	41519	10.252.174.188	22	13	4 kB	5	26	50.00%	7	
23.20.23.147	41634	10.252.174.188	22	13	4 kB	9	26	50.00%	7	
23.20.23.147	42125	10.252.174.188	22	13	4 kB	48	26	50.00%	7	
23.20.23.147	42574	10.252.174.188	22	13	4 kB	1	26	50.00%	7	
23.20.23.147	42691	10.252.174.188	22	13	4 kB	37	26	50.00%	7	
23.20.23.147	43457	10.252.174.188	22	13	4 kB	13	26	50.00%	7	
23.20.23.147	43507	10.252.174.188	22	13	4 kB	16	26	50.00%	7	
23.20.23.147	43935	10.252.174.188	22	13	4 kB	40	26	50.00%	7	
23.20.23.147	44640	10.252.174.188	22	13	4 kB	41	26	50.00%	7	
23.20.23.147	44907	10.252.174.188	22	13	4 kB	50	26	50.00%	7	
23.20.23.147	45869	10.252.174.188	22	13	4 kB	11	28	46.43%	7	
23.20.23.147	47260	10.252.174.188	22	13	4 kB	29	26	50.00%	7	
23.20.23.147	47447	10.252.174.188	22	13	4 kB	7	26	50.00%	7	
23.20.23.147	47769	10.252.174.188	22	13	4 kB	25	26	50.00%	7	
23.20.23.147	47940	10.252.174.188	22	13	4 kB	51	26	50.00%	7	
23.20.23.147	49484	10.252.174.188	22	13	4 kB	4	26	50.00%	7	
23.20.23.147	49744	10.252.174.188	22	13	4 kB	33	26	50.00%	7	
23.20.23.147	51055	10.252.174.188	22	13	4 kB	49	26	50.00%	7	
23.20.23.147	51492	10.252.174.188	22	13	4 kB	2	26	50.00%	7	
23.20.23.147	51803	10.252.174.188	22	13	4 kB	14	28	46.43%	7	
23.20.23.147	52014	10.252.174.188	22	13	4 kB	24	26	50.00%	7	
23.20.23.147	52474	10.252.174.188	22	13	4 kB	28	26	50.00%	7	
23.20.23.147	52758	10.252.174.188	22	13	4 kB	42	26	50.00%	7	
23.20.23.147	53469	10.252.174.188	22	13	4 kB	30	26	50.00%	7	
23.20.23.147	53534	10.252.174.188	22	13	4 kB	45	26	50.00%	7	
23.20.23.147	55025	10.252.174.188	22	13	4 kB	3	26	50.00%	7	
23.20.23.147	57384	10.252.174.188	22	13	4 kB	23	26	50.00%	7	
23.20.23.147	57752	10.252.174.188	22	13	4 kB	12	26	50.00%	7	
23.20.23.147	57787	10.252.174.188	22	13	4 kB	47	26	50.00%	7	
23.20.23.147	58283	10.252.174.188	22	13	4 kB	46	26	50.00%	7	
23.20.23.147	58441	10.252.174.188	22	13	4 kB	39	26	50.00%	7	
23.20.23.147	58563	10.252.174.188	22	13	4 kB	32	26	50.00%	7	
23.20.23.147	58975	10.252.174.188	22	13	4 kB	8	26	50.00%	7	
23.20.23.147	59034	10.252.174.188	22	13	4 kB	27	26	50.00%	7	
23.20.23.147	59095	10.252.174.188	22	13	4 kB	22	26	50.00%	7	
23.20.23.147	59172	10.252.174.188	22	13	4 kB	15	27	48.15%	7	
23.20.23.147	59454	10.252.174.188	22	13	4 kB	35	26	50.00%	7	
23.20.23.147	60067	10.252.174.188	22	13	4 kB	44	26	50.00%	7	
23.20.23.147	60670	10.252.174.188	22	13	4 kB	38	26	50.00%	7	

To identify the number of failed attempts, make sure that you are still filtering for SSH traffic in the main Wireshark view.

Navigate to "*Statistics > Conversations > TCP tab*" in Wireshark. At the bottom of the conversations window, there is a checkbox option to limit what we see to our display filter only (i.e. SSH traffic). After enabling this option, we see only SSH traffic under the TCP tab. Check the number. See how many were successful and reduce it from total attempts the attacker made.


Wireshark · Conversations · hp_challenge.pcap
Copy
Follow Stream...
Graph...
Filter list for specific type
Help
Close

Conversation Settings

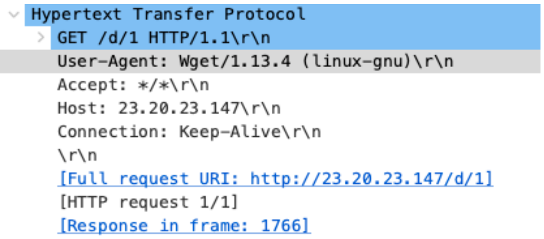
☐ Name resolution
☐ Absolute start time
☑ Limit to display filter

Protocol
☐ Bluetooth
☐ DCCP
☑ Ethernet
☐ FC
☐ FDDI
☐ IEEE 802.11

cent Filtered	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bi
55.34%	38	6 kB	221	29 kB	114.538382	12.1979	4
68.00%	13	2 kB	21	5 kB	109.783033	0.7182	
50.00%	7	1 kB	6	2 kB	2.609629	2.6149	42
48.15%	7	1 kB	6	2 kB	32.549862	2.4366	45
50.00%	7	1 kB	6	2 kB	34.986513	2.2395	49
50.00%	7	1 kB	6	2 kB	41.842225	2.4816	4
50.00%	7	1 kB	6	2 kB	47.002385	1.6069	69
50.00%	7	1 kB	6	2 kB	48.610028	2.0830	53
50.00%	7	1 kB	6	2 kB	50.693905	2.3504	47
50.00%	7	1 kB	6	2 kB	53.045278	2.0996	52
50.00%	7	1 kB	6	2 kB	55.141688	2.0411	54
50.00%	7	1 kB	6	2 kB	58.824057	1.9158	5
50.00%	7	1 kB	6	2 kB	63.132721	1.4668	75
50.00%	7	1 kB	6	2 kB	72.656782	2.3700	4
50.00%	7	1 kB	6	2 kB	75.027512	1.4445	7
50.00%	7	1 kB	6	2 kB	87.046496	1.4255	77
50.00%	7	1 kB	6	2 kB	88.472725	1.8984	58
50.00%	7	1 kB	6	2 kB	105.708068	1.8990	58
50.00%	7	1 kB	6	2 kB	0.000000	0.2649	

6. (10 Points) What is the tool used to download malicious files on the system?
   The tool used to download the malicious fils on the system can be seen in the screenshot below, it was using wget, which is a tool that allows user to download files on HTTP, HTTPS, and FTP.
   Based on our earlier findings, we know that the only other protocol present, apart from SSH, is HTTP. Now, filter the HTTP traffic.

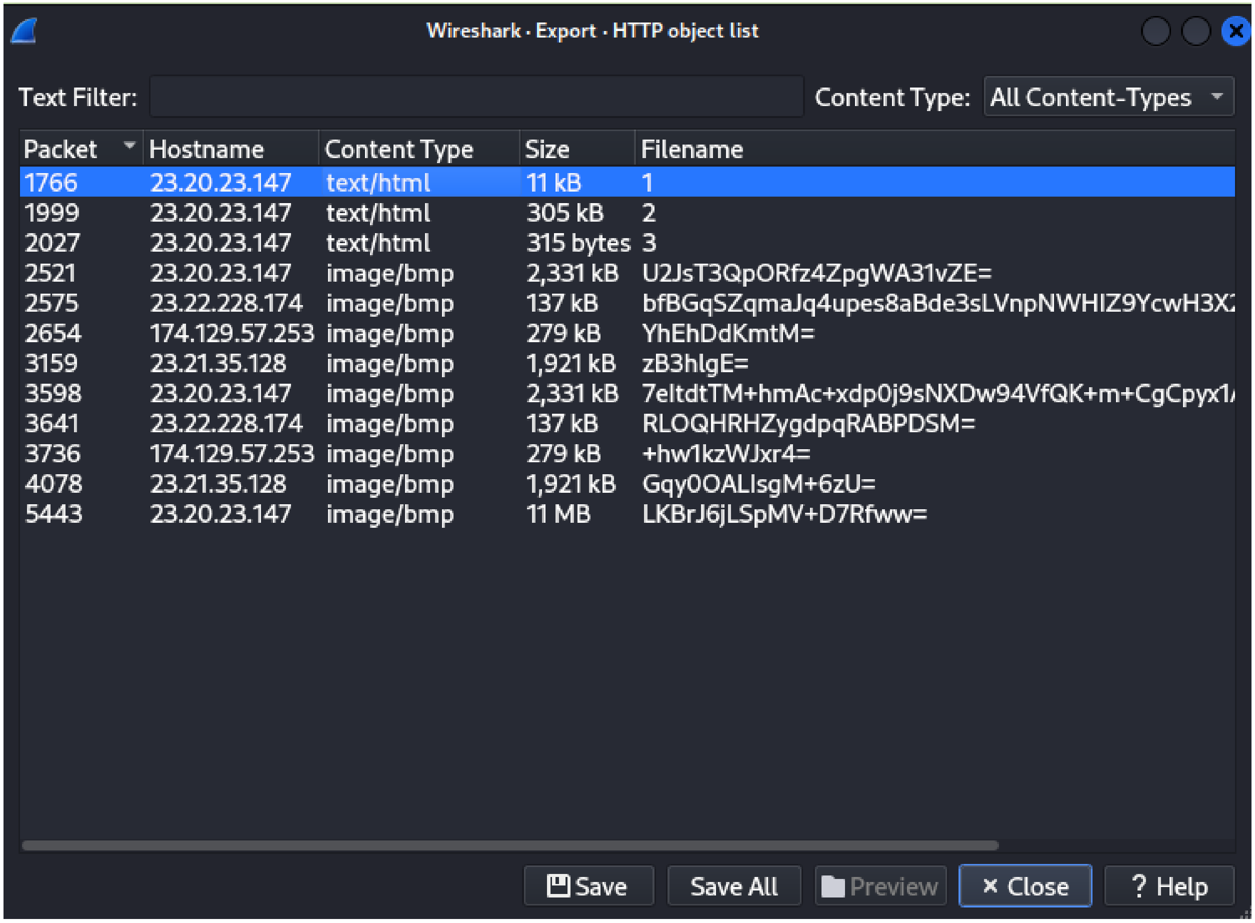
   Select the first HTTP packet and follow it's HTTP stream. If you look carefully at the request headers highlighted, you can see that the User-Agent request header has the value. It is a tool that retrieves content from web servers by downloading via HTTP, HTTPS, and FTP.



```
Hypertext Transfer Protocol
  > GET /d/1 HTTP/1.1\r\n
    User-Agent: Wget/1.13.4 (linux-gnu)\r\n
    Accept: */*\r\n
    Host: 23.20.23.147\r\n
    Connection: Keep-Alive\r\n
    \r\n
    [Full request URI: http://23.20.23.147/d/1]
    [HTTP request 1/1]
    [Response in frame: 1766]
```

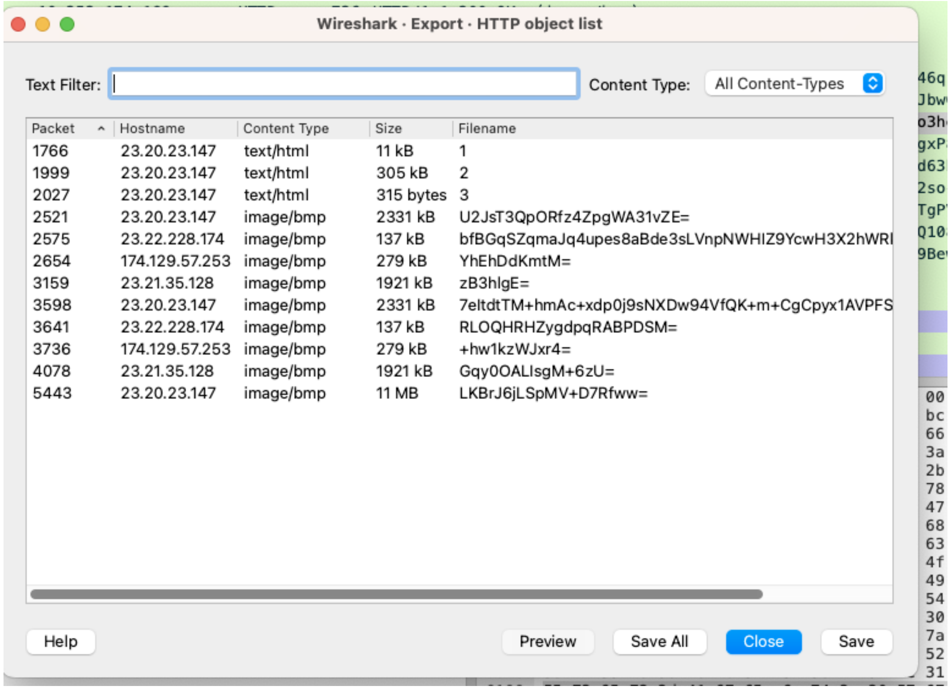
7. (10 Points) Find out how many files the attacker downloaded to perform malware installation. Attach screenshot.

Below is the screenshot from my Wireshark export. There are 12 files total including filename 1, 2, and 3 which could be part of the attack. According to google, typically malicious attackers install malware with base64 encoded names to obfuscate the file names of their attack which would be the bottom 9 files. The BMP files indicate the malware installation, so the attacker downloaded 9 files.


Wireshark · Export · HTTP object list
Text Filter:
Content Type: All Content-Types
Save
Save All
Preview
× Close
? Help

Packet	Hostname	Content Type	Size	Filename
1766	23.20.23.147	text/html	11 kB	1
1999	23.20.23.147	text/html	305 kB	2
2027	23.20.23.147	text/html	315 bytes	3
2521	23.20.23.147	image/bmp	2,331 kB	U2JsT3QpORfz4ZpgWA31vZE=
2575	23.22.228.174	image/bmp	137 kB	bfBGqSZqmaJq4upes8aBde3sLVnpNWHIZ9YcwH3X
2654	174.129.57.253	image/bmp	279 kB	YhEhDdKmtM=
3159	23.21.35.128	image/bmp	1,921 kB	zB3hlgE=
3598	23.20.23.147	image/bmp	2,331 kB	7eltdtTM+hmAc+xdp0j9sNXDw94VfQK+m+CgCpyx1
3641	23.22.228.174	image/bmp	137 kB	RLOQHRHZygdpqRABPDSM=
3736	174.129.57.253	image/bmp	279 kB	+hw1kzWJxr4=
4078	23.21.35.128	image/bmp	1,921 kB	Gqy0OALIsgM+6zU=
5443	23.20.23.147	image/bmp	11 MB	LKBrJ6jLSpMV+D7Rfww=

To answer this question, navigate to "*File > Export Objects > HTTP*" in Wireshark. In this window, you can see three files named **1**, **2** and **3**. There are also multiple BMP files with base64 encoded filenames:

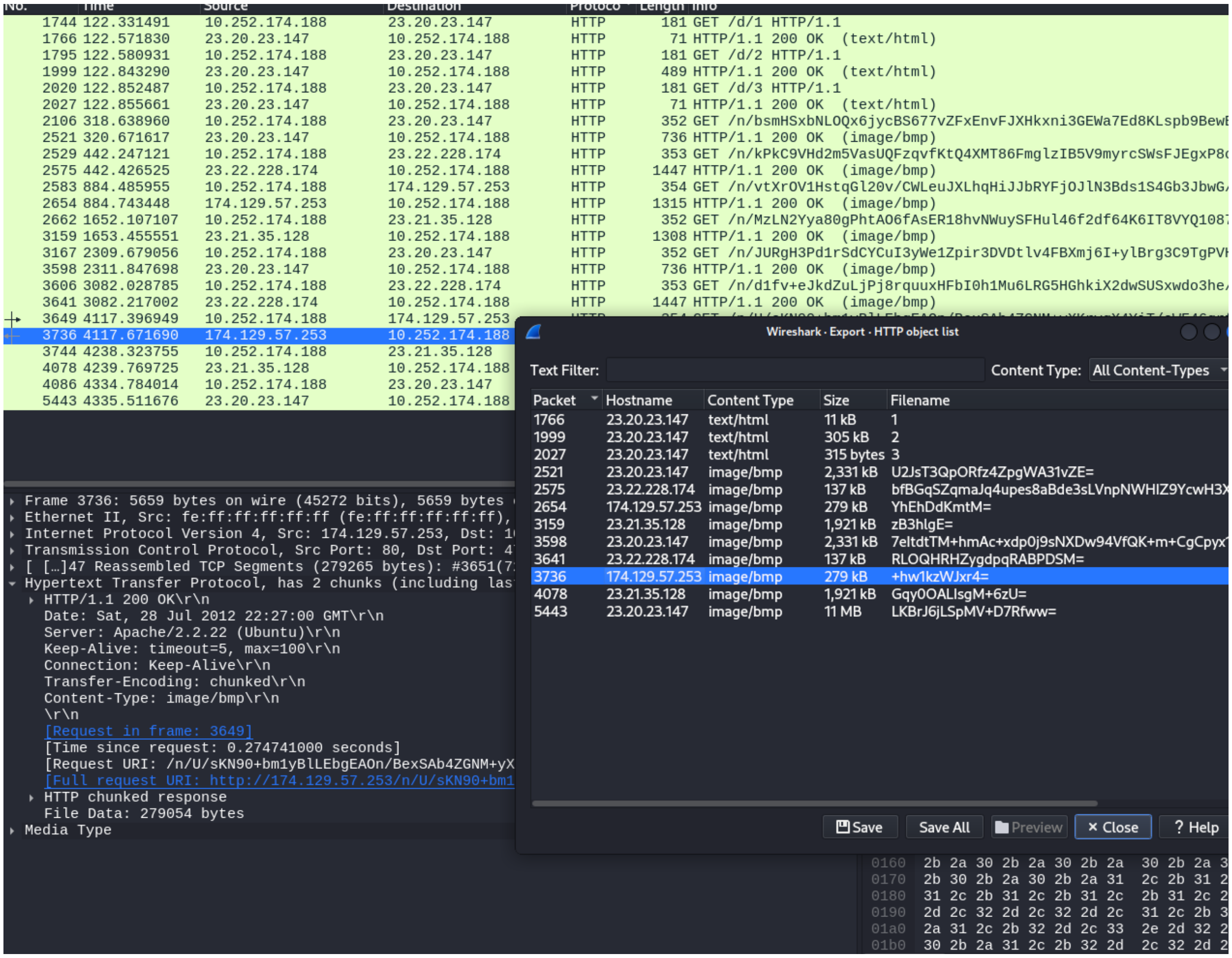

Wireshark · Export · HTTP object list
Text Filter:
Content Type: All Content-Types
Help
Preview
Save All
Close
Save

Packet	Hostname	Content Type	Size	Filename
1766	23.20.23.147	text/html	11 kB	1
1999	23.20.23.147	text/html	305 kB	2
2027	23.20.23.147	text/html	315 bytes	3
2521	23.20.23.147	image/bmp	2331 kB	U2JsT3QpORfz4ZpgWA31vZE=
2575	23.22.228.174	image/bmp	137 kB	bfBGqSZqmaJq4upes8aBde3sLVnpNWHIZ9YcwH3X2hWRI
2654	174.129.57.253	image/bmp	279 kB	YhEhDdKmtM=
3159	23.21.35.128	image/bmp	1921 kB	zB3hlgE=
3598	23.20.23.147	image/bmp	2331 kB	7eltdtTM+hmAc+xdp0j9sNXDw94VfQK+m+CgCpyx1AVPFS
3641	23.22.228.174	image/bmp	137 kB	RLOQHRHZygdpqRABPDSM=
3736	174.129.57.253	image/bmp	279 kB	+hw1kzWJxr4=
4078	23.21.35.128	image/bmp	1921 kB	Gqy0OALIsgM+6zU=
5443	23.20.23.147	image/bmp	11 MB	LKBrJ6jLSpMV+D7Rfww=

HTTP Objects

Copy your screenshot here and label it.

8. (10 Points) One of the IP's the malware contacted starts with 17. Find and provide the full IP.
   174.129.57.253
   (screenshot below)



```
No.   Time          Source          Destination     Protoco  Length Info
 1744 122.331491    10.252.174.188  23.20.23.147    HTTP      181 GET /d/1 HTTP/1.1
 1766 122.571830    23.20.23.147    10.252.174.188  HTTP       71 HTTP/1.1 200 OK  (text/html)
 1795 122.580931    10.252.174.188  23.20.23.147    HTTP      181 GET /d/2 HTTP/1.1
 1999 122.843290    23.20.23.147    10.252.174.188  HTTP      489 HTTP/1.1 200 OK  (text/html)
 2020 122.852487    10.252.174.188  23.20.23.147    HTTP      181 GET /d/3 HTTP/1.1
 2027 122.855661    23.20.23.147    10.252.174.188  HTTP       71 HTTP/1.1 200 OK  (text/html)
 2106 318.638960    10.252.174.188  23.20.23.147    HTTP      352 GET /n/bsmHSxbNLOQx6jycBS677vZFxEnvFJXHkxni3GEWa7Ed8KLspb9BewE
 2521 320.671617    23.20.23.147    10.252.174.188  HTTP      736 HTTP/1.1 200 OK  (image/bmp)
 2529 442.247121    10.252.174.188  23.22.228.174   HTTP      353 GET /n/kPkC9VHd2m5VasUQFzqvfKtQ4XMT86FmglzIB5V9myrcSWsFJEgxP8c
 2575 442.426525    23.22.228.174   10.252.174.188  HTTP     1447 HTTP/1.1 200 OK  (image/bmp)
 2583 884.485955    10.252.174.188  174.129.57.253  HTTP      354 GET /n/vtXrOV1HstqGl20v/CWLeuJXLhqHiJJbRYFjOJlN3Bds1S4Gb3JbwG/
 2654 884.743448    174.129.57.253  10.252.174.188  HTTP     1315 HTTP/1.1 200 OK  (image/bmp)
 2662 1652.107107   10.252.174.188  23.21.35.128    HTTP      352 GET /n/MzLN2Yya80gPhtAO6fAsER18hvNWuySFHul46f2df64K6IT8VYQ108
 3159 1653.455551   23.21.35.128    10.252.174.188  HTTP     1308 HTTP/1.1 200 OK  (image/bmp)
 3167 2309.679056   10.252.174.188  23.20.23.147    HTTP      352 GET /n/JURgH3Pd1rSdCYCuI3yWe1Zpir3DVDtlv4FBXmj6I+ylBrg3C9TgPVH
 3598 2311.847698   23.20.23.147    10.252.174.188  HTTP      736 HTTP/1.1 200 OK  (image/bmp)
 3606 3082.028785   10.252.174.188  23.22.228.174   HTTP      353 GET /n/d1fv+eJkdZuLjPj8rquuxHFbI0h1Mu6LRG5HGhkiX2dwSUSxwdo3he/
 3641 3082.217002   23.22.228.174   10.252.174.188  HTTP     1447 HTTP/1.1 200 OK  (image/bmp)
 3649 4117.396949   10.252.174.188  174.129.57.253
 3736 4117.671690   174.129.57.253  10.252.174.188
 3744 4238.323755   10.252.174.188  23.21.35.128
 4078 4239.769725   23.21.35.128    10.252.174.188
 4086 4334.784014   10.252.174.188  23.20.23.147
 5443 4335.511676   23.20.23.147    10.252.174.188
```

```
▸ Frame 3736: 5659 bytes on wire (45272 bits), 5659 bytes
▸ Ethernet II, Src: fe:ff:ff:ff:ff:ff (fe:ff:ff:ff:ff:ff),
▸ Internet Protocol Version 4, Src: 174.129.57.253, Dst: 1
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 4
▸ [ […]47 Reassembled TCP Segments (279265 bytes): #3651(7
▾ Hypertext Transfer Protocol, has 2 chunks (including las
  ▸ HTTP/1.1 200 OK\r\n
    Date: Sat, 28 Jul 2012 22:27:00 GMT\r\n
    Server: Apache/2.2.22 (Ubuntu)\r\n
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Transfer-Encoding: chunked\r\n
    Content-Type: image/bmp\r\n
    \r\n
    [Request in frame: 3649]
    [Time since request: 0.274741000 seconds]
    [Request URI: /n/U/sKN90+bm1yBlLEbgEAOn/BexSAb4ZGNM+yX
    [Full request URI: http://174.129.57.253/n/U/sKN90+bm1
  ▸ HTTP chunked response
    File Data: 279054 bytes
▸ Media Type
```

Text Filter:

Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
1766	23.20.23.147	text/html	11 kB	1
1999	23.20.23.147	text/html	305 kB	2
2027	23.20.23.147	text/html	315 bytes	3
2521	23.20.23.147	image/bmp	2,331 kB	U2JsT3QpORfz4ZpgWA31vZE=
2575	23.22.228.174	image/bmp	137 kB	bfBGqSZqmaJq4upes8aBde3sLVnpNWHIZ9YcwH3X
2654	174.129.57.253	image/bmp	279 kB	YhEhDdKmtM=
3159	23.21.35.128	image/bmp	1,921 kB	zB3hlgE=
3598	23.20.23.147	image/bmp	2,331 kB	7eltdtTM+hmAc+xdp0j9sNXDw94VfQK+m+CgCpyx
3641	23.22.228.174	image/bmp	137 kB	RLOQHRHZygdpqRABPDSM=
3736	174.129.57.253	image/bmp	279 kB	+hw1kzWJxr4=
4078	23.21.35.128	image/bmp	1,921 kB	Gqy0OALIsgM+6zU=
5443	23.20.23.147	image/bmp	11 MB	LKBrJ6jLSpMV+D7Rfww=

```
0160  2b 2a 30 2b 2a 30 2b 2a  30 2b 2a 3
0170  2b 30 2b 2a 30 2b 2a 31  2c 2b 31 2
0180  31 2c 2b 31 2c 2b 31 2c  2b 31 2c 2
0190  2d 2c 32 2d 2c 32 2d 2c  31 2c 2b 3
01a0  2a 31 2c 2b 32 2d 2c 33  2e 2d 32 2
01b0  30 2b 2a 31 2c 2b 32 2d  2c 32 2d 2
```

Refer back to "*File > Export Objects > HTTP*", where we can see the IP address that starts with **17** that was contacted by the malware to download BMP files.

9. (30 Points) Short answers (2-3 sentences).
    a. What do you think are the potential ethical and legal considerations when using Wireshark for packet capture?

I think that Wireshark is a very powerful tool which can be used to intercept traffic on a network. It can intercept traffic of communication or even credentials which violates privacy laws and is ethically wrong. Unauthorized packet capture can be illegal in situations which delve into very sensitive information so when doing so should be done with clear authorization for all parties involved.

    b. Define the term "packet analysis" and explain its importance in network troubleshooting and security.

Packet analysis is important as it is used to capture, inspect, analyze, and interpret the movement of data between two different connections. Being able to understand, and identify ports and the movement of data is important to learn how to protect that data from being exposed and to ensure that there aren't any unauthorized users accessing data with malicious activities.

    c. List some signs or network traffic behaviors that may indicate the presence of malware.

Unusual outbound traffic with large data transfers to unknown IP's That could be suspicious. The utilization of ports on the network that are uncommon, high volume of DNS requests, encrypted traffic spikes, and repetitive connection attempts

d. What is a brute force attack, and how does it work?

A brute force attack is a type of attack which utilizes trial and error method by attackers to gain unauthorized access to a system by try all possible combinations of credentials of usernames and passwords. It keeps trying until the correct one is found and typically sends login requests to a target service like SSH or RDP usually using dictionary attacks.

e. What are some common strategies to defend against brute force attacks based on the information obtained from packet captures?

Probably one of the easiest ways to counter brute force attack is by rate limiting the amount of successive login attempts. Forcing users to wait just up to 5 seconds between logins significantly slows down the process for brute force attacks. Lockout policies for unsuccessful logins as well would work too. MFA is drastically helps too.

f. How can Wireshark help in detecting and analyzing brute force attacks on network services like SSH or RDP?

Wireshark can detect brute force attacks on services like SSH or RDP by capturing the network traffic and analyzing it for signs of high volume repeated login attempts from a single IP. Wireshark can identify the attackers IP and filter packets for tcp (port 22) and RDP (port 3389) as well for failed authentication responses or unusual packet rates indicated an attack.

Submission Instructions:

- Submit the screenshots and filled pdf of this document (with all the answers). Do not delete the questions or change the order of the questions. You can download and edit this document.
- Submit electronically through Canvas.
- Email or hardcopy submissions will not be accepted.

References:
https://www.comparitech.com/net-admin/pcap-guide/#:~:text=To%20capture%20PCAP%20files%20you,could%20be%20eth0%20or%20wlan0
This lab is Inspired from EscapeRoom CTF created by *The Honeynet Project* on the CyberDefenders website:
https://cyberdefenders.org/blueteam-ctf-challenges/18#nav-overview