

Wireshark

Analyzing 2 pcap files in wireshark

PCAP 1

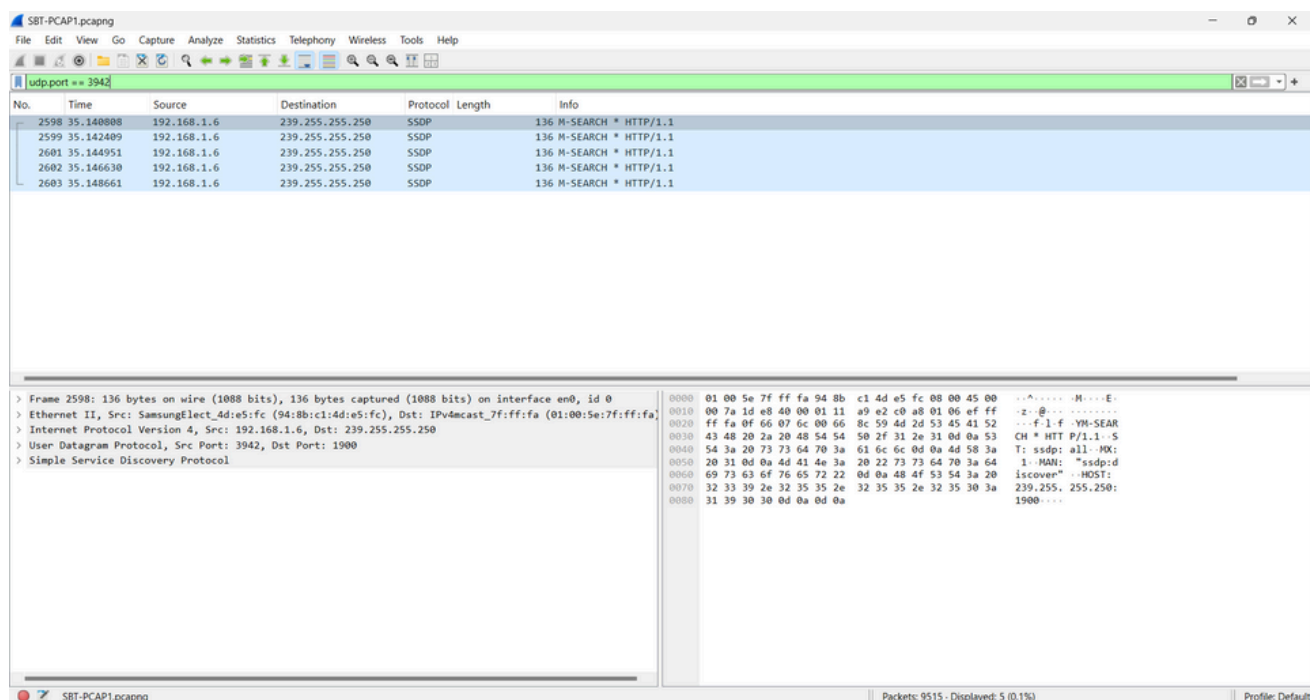
1. Which protocol was used over port 3942?
2. What is the IP address of the host that was pinged twice?
3. How many DNS query response packets were captured?
4. What is the IP address of the host which sent the most number of bytes?

PCAP 2

1. What is the WebAdmin password?
2. What is the version number of the attacker's FTP server?
3. Which port was used to gain access to the victim Windows host?
4. What is the name of a confidential file on the Windows host?
5. What is the name of the log file that was created at 4:51 AM on the Windows host?

1. Protocol Used Over Port 3942

To identify the protocol, I applied the Wireshark filter `udp.port == 3942`. By doing this, I observed that the protocol being used over port 3942 was **SSDP (Simple Service Discovery Protocol)**.



2. IP Address of Host Pinged Twice

I used the icmp filter and looked specifically for "echo request" packets. Through this analysis, I identified that the IP address **8.8.4.4** was the host that was pinged twice.

SBT-PCAP1.pcapng						
icmp						
No.	Time	Source	Destination	Protocol	Length	Info
127	4.331987	192.168.1.7	8.8.8.8	ICMP	70	Destination unreachable (Port unreachable)
128	4.331987	192.168.1.7	8.8.8.8	ICMP	70	Destination unreachable (Port unreachable)
1632	22.769823	192.168.1.7	8.8.4.4	ICMP	98	Echo (ping) request id=0x4728, seq=0/0, ttl=64 (reply in 1665)
1665	23.353519	8.8.4.4	192.168.1.7	ICMP	98	Echo (ping) reply id=0x4728, seq=0/0, ttl=56 (request in 1632)
1671	23.774920	192.168.1.7	8.8.4.4	ICMP	98	Echo (ping) request id=0x4728, seq=1/256, ttl=64 (reply in 1708)
1708	24.477631	8.8.4.4	192.168.1.7	ICMP	98	Echo (ping) reply id=0x4728, seq=1/256, ttl=56 (request in 1671)

> Frame 1632: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, id 0

> Ethernet II, Src: Apple_93:ad:f0 (8c:85:90:93:ad:f0), Dst: Netcomm_8f:82:2b (00:60:64:8f:82:2b)

> Internet Protocol Version 4, Src: 192.168.1.7, Dst: 8.8.4.4

> Internet Control Message Protocol

0000 00 60 64 8f 82 2b 8c 85 90 93 ad f0 08 00 45 00d...E

0010 00 54 64 a3 00 00 40 01 48 4b c0 a8 01 07 08 08 ...Td...@...HK.....

0020 04 04 08 00 1f ae 47 28 00 00 5e 2f c5 98 00 08G(.....

0030 82 56 88 09 0a 00 0c 0d 0e 0f 10 11 12 13 14 15V.....

0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25l...\$

0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35~()*+.../012345

0060 36 37

3. Number of DNS Query Response Packets Captured

To count the DNS query response packets, I applied the Wireshark display filter dns.flags.response == True. This filter specifically shows packets that are DNS responses. The analysis revealed that there were **90** such packets captured in the PCAP file.

SBT-PCAP1.pcapng						
dns.flags.response == True						
No.	Time	Source	Destination	Protocol	Length	Info
833	13.159359	8.8.8.8	192.168.1.7	DNS	91	Standard query response 0xd0dd A ssl.gstatic.com A 172.217.25.163
835	13.163264	8.8.8.8	192.168.1.7	DNS	93	Standard query response 0x92c6 A plus.l.google.com A 216.58.199.46
929	14.383934	8.8.8.8	192.168.1.7	DNS	138	Standard query response 0x2325 A au.yahoo.com CNAME atsv2-fp-shed.wg1.b.yahoo.com A 106.10.250.11 A 106.10.250.10
987	15.638814	8.8.8.8	192.168.1.7	DNS	141	Standard query response 0x6773 A s.yimg.com CNAME edge.gycpi.b.yahoodns.net A 115.178.9.18 A 115.178.9.19
988	15.638818	8.8.8.8	192.168.1.7	DNS	142	Standard query response 0xc3f2 No such name A csc.beap.bc.yahoo.com SOA ns1.yahoo.com
989	15.639464	8.8.8.8	192.168.1.7	DNS	144	Standard query response 0x856d A geo.yahoo.com CNAME fam-geo-atsv2.prod.media.g03.yahoodns.net A 98.136.103.27
990	15.639469	8.8.8.8	192.168.1.7	DNS	136	Standard query response 0x84a1 A comet.yahoo.com CNAME ds-comet.yahoo.g01.yahoodns.net A 106.10.218.138
991	15.643308	8.8.8.8	192.168.1.7	DNS	161	Standard query response 0x427c A video-api.yql.yahoo.com CNAME geocpi-uno.gycpi.b.yahoodns.net A 115.178.9.19 A 115.178.9.18
1086	16.228235	8.8.8.8	192.168.1.7	DNS	98	Standard query response 0xa0d2 A dart.l.doubleclick.net A 172.217.167.70
2100	29.779492	8.8.8.8	192.168.1.7	DNS	200	Standard query response 0x8429 A guce.yahoo.com CNAME real.rotation.guce.aws.oath.cloud CNAME prod-rotation-v2.guce.aws.oath.cloud
2292	31.871315	8.8.8.8	192.168.1.7	DNS	103	Standard query response 0xf07d A mdkodai.com A 104.28.22.97 A 104.28.23.97
2351	32.542995	8.8.8.8	192.168.1.7	DNS	135	Standard query response 0x20bd A login.yahoo.com CNAME ds-ats.member.g02.yahoodns.net A 74.6.160.138
2352	32.545684	8.8.8.8	192.168.1.7	DNS	145	Standard query response 0x536c A mail.yahoo.com CNAME edge.gycpi.b.yahoodns.net A 115.178.9.18 A 115.178.9.19
2361	32.655205	8.8.8.8	192.168.1.7	DNS	138	Standard query response 0xdffa A au.search.yahoo.com CNAME ds-global3.l7.search.ystg1.b.yahoo.com A 106.10.218.137
2362	32.662304	8.8.8.8	192.168.1.7	DNS	197	Standard query response 0xb75e A au.news.yahoo.com CNAME media-router1.prod.media.yahoo.com CNAME ds-oob-fo-media-router1.prod.media.yahoo.com
2365	32.689427	8.8.8.8	192.168.1.7	DNS	202	Standard query response 0x4651 A aka-cdn.adtechus.com CNAME cs704.wpc.thetacdn.net CNAME cs701.lb.wpc.apr-1b09e.edgecdn.adtechus.com
2460	33.604606	8.8.8.8	192.168.1.7	DNS	202	Standard query response 0xdd62 A au.lifestyle.yahoo.com CNAME media-router1.prod.media.yahoo.com CNAME ds-oob-fo-media-router1.prod.media.yahoo.com

> Frame 1086: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, id 0

> Ethernet II, Src: Netcomm_8f:82:2b (00:60:64:8f:82:2b), Dst: Apple_93:ad:f0 (8c:85:90:93:ad:f0)

> Internet Protocol Version 4, Src: 8.8.8.8, Dst: 192.168.1.7

> User Datagram Protocol, Src Port: 53, Dst Port: 64068

> Domain Name System (response)

Transaction ID: 0xa0d2

Flags: 0x8180 Standard query response, No error

1... .. = Response: Message is a response

.000 0... .. = Opcode: Standard query (0)

... ..0... .. = Authoritative: Server is not an authority for domain

... ..0... .. = Truncated: Message is not truncated

... ..1... .. = Recursion desired: Do query recursively

... ..1... .. = Recursion available: Server can do recursive queries

... ..0... .. = Z: reserved (0)

... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server

... ..0... .. = Non-authenticated data: Unacceptable

... ..0000 = Reply code: No error (0)

Questions: 1

0000 8c 85 90 93 ad f0 00 60 64 8f 82 2b 08 00 45 00d...E

0010 00 54 82 72 00 00 7b 11 eb 67 08 08 08 c0 a8 ...T...{...g.....

0020 01 07 00 35 fa 44 00 40 55 b9 a0 d2 81 80 00 01 ...5-D...@...U.....

0030 00 01 00 00 00 04 64 61 72 74 01 6c 0b 64 0fd...art...l...do

0040 75 62 6c 65 63 6c 69 63 6b 03 6e 65 74 00 00 01 ubliclic k-net...

0050 00 01 c0 0c 00 01 00 01 00 00 0e 00 04 ac d9

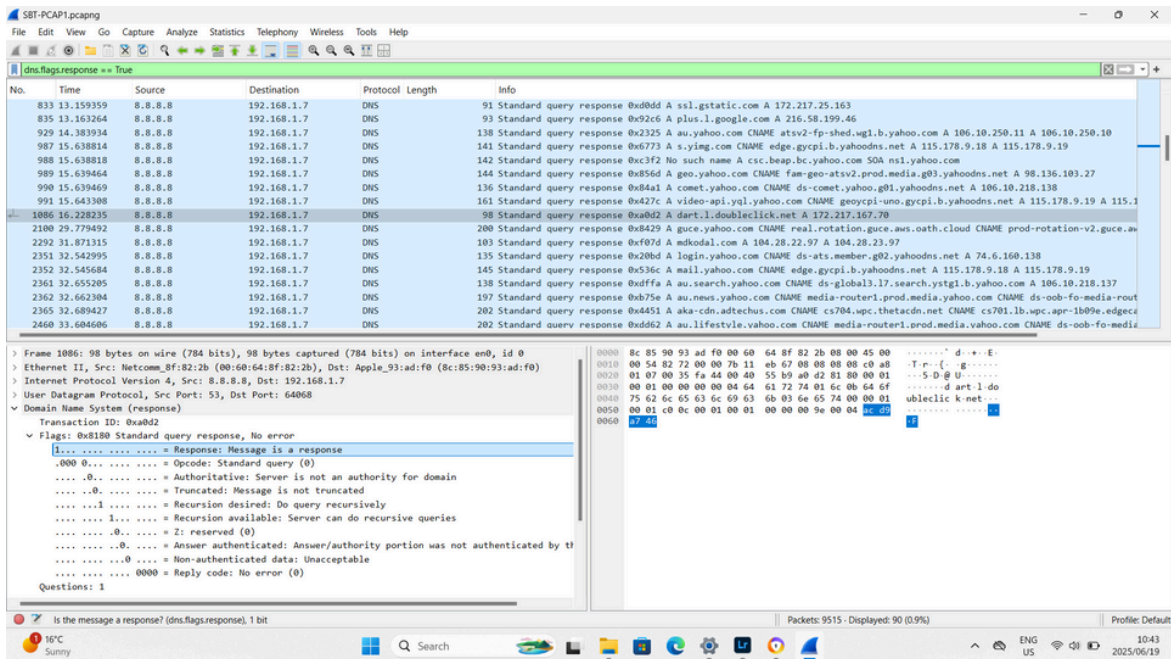
0060 a7 46

Is the message a response? (dns.flags.response), 1 bit

Packets: 9515 - Displayed: 90 (0.9%)

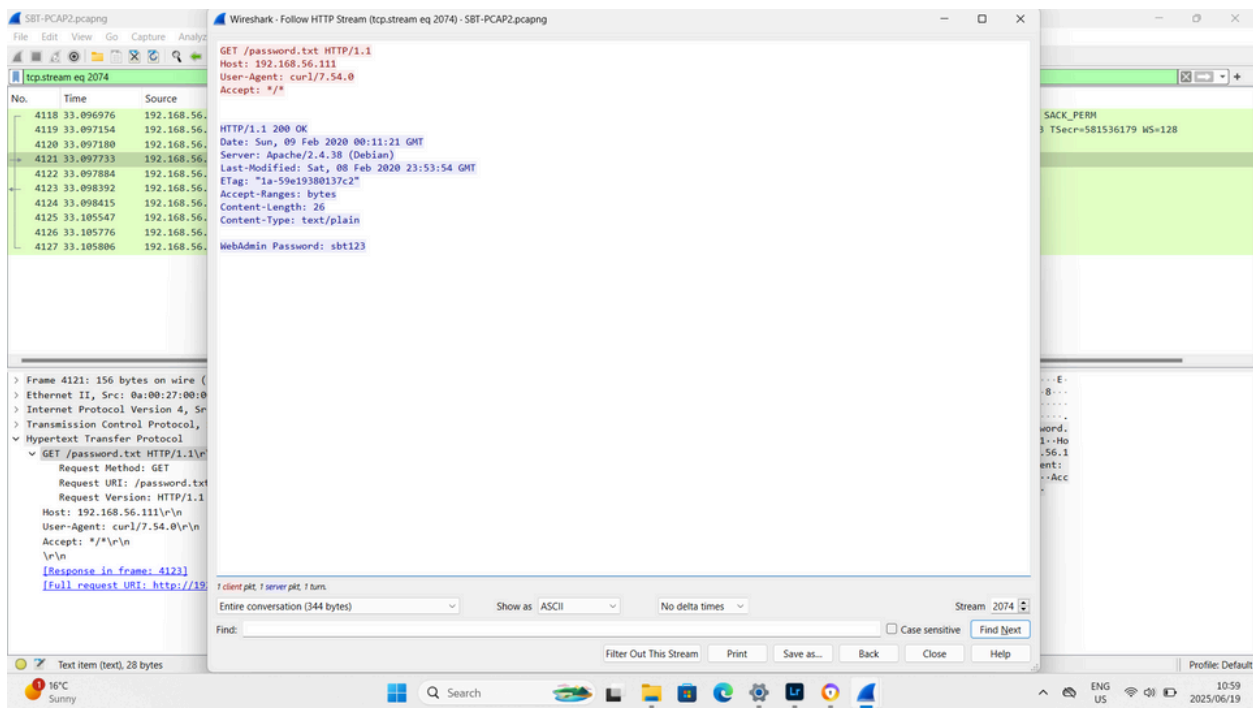
Profile: Default

4. To find the IP address of the host that sent the most bytes, I navigated to the Statistics menu, then selected Endpoints, and specifically the IPv4 tab. From there, I clicked on Tx Bytes (transmit bytes) to arrange the results in descending order. The IP address that sent the most data was **115.178.9.18**.



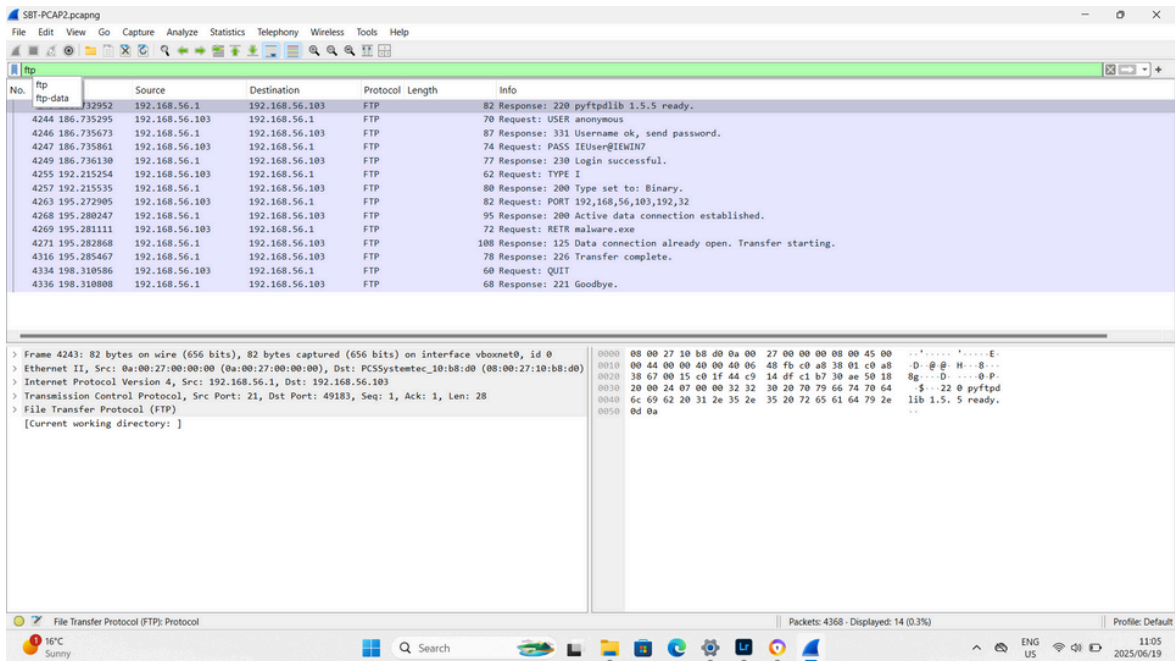
1. WebAdmin Password

To find the WebAdmin password in PCAP 2, I started by applying an http filter in Wireshark. I then looked for packets that used the GET method. After selecting the relevant packet, I clicked on the "Hypertext Transfer Protocol" details and chose to "Follow HTTP Stream." This allowed me to see the full conversation. Through this stream, I successfully found the WebAdmin password, which was **sbt123**.



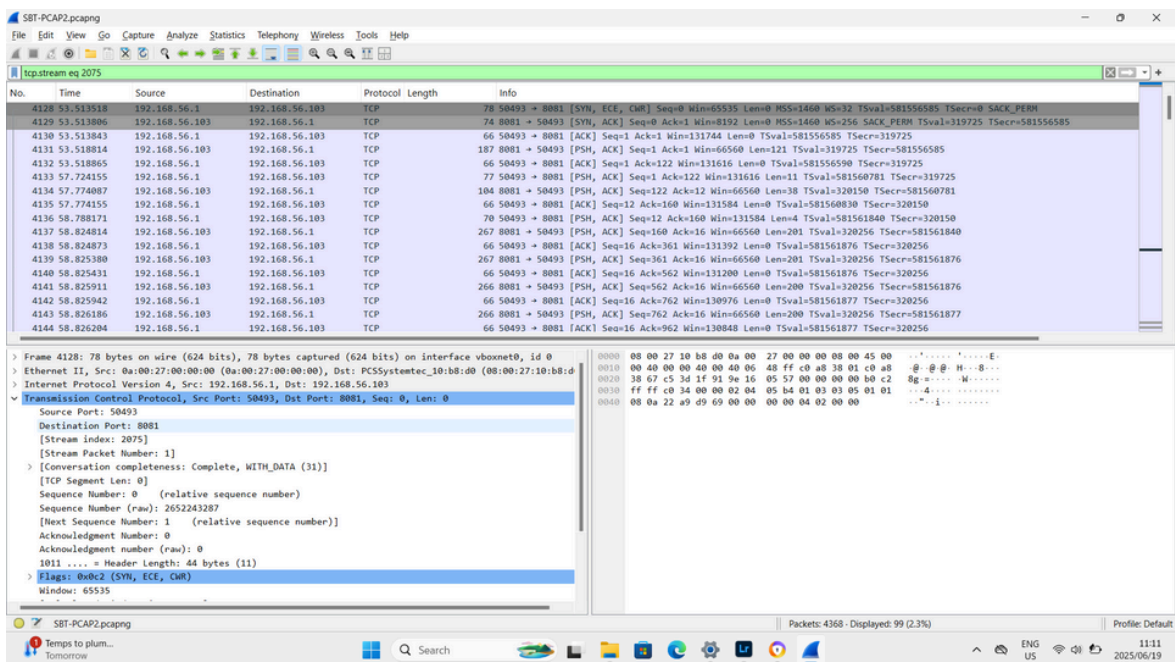
2. Attacker's FTP Server Version

To determine the version number of the attacker's FTP server, I applied an ftp filter in Wireshark. I then examined the information contained within the first FTP packet. This packet revealed that the FTP server version was pyftplib **1.5.5**.



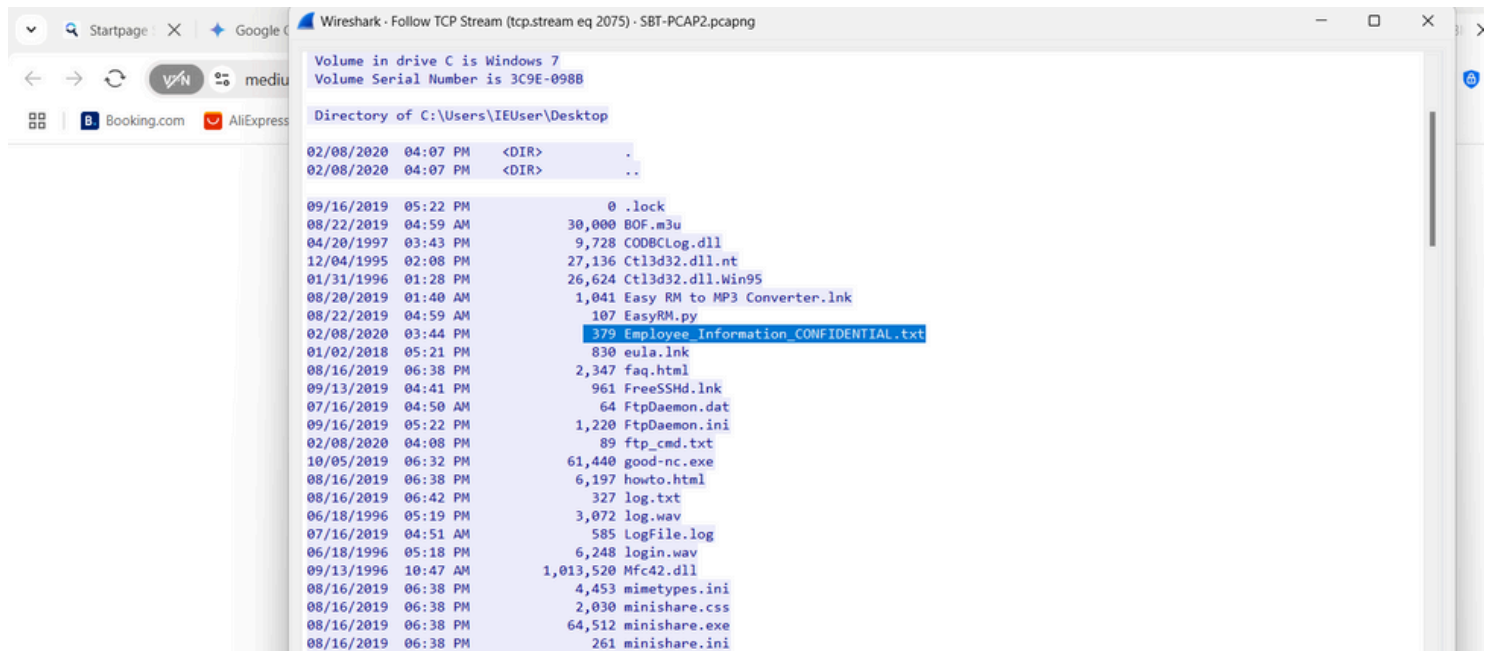
3. Port Used to Gain Access to Victim Windows Host

To identify the port used to gain access to the victim's Windows host, I applied a filter for ip.src == 192.168.56.1. After identifying relevant packets, I clicked packet 4128 then examined the "Transmission Control Protocol" panel. There, I found that the destination port used for access was **8081**. This port is often used for alternate HTTP services or web management interfaces.



4. Name of Confidential File on Windows Host

Continuing my analysis on packet 4128, I followed its TCP stream. Within this stream, I discovered a highly sensitive file named **Employee_Information_CONFIDENTIAL.txt**. This file likely contains private employee data and should not have been accessible or transferred in this manner



5. Name of Log File Created at 4:51 AM on Windows Host

Still within the same TCP stream from packet 4128, I looked for evidence of log file creation. I successfully found a log file named **LogFile.log**. This file was created on July 16, 2019, at 4:51 AM,

