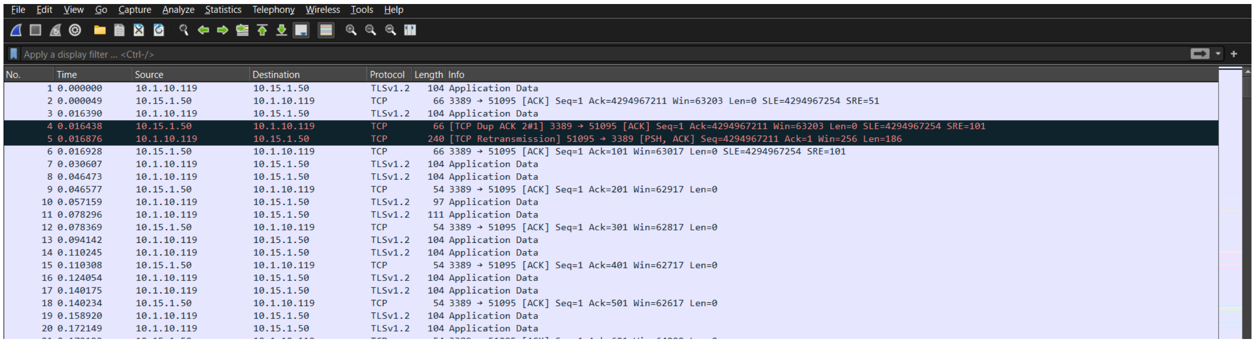
**Description:** The goal of this project is to demonstrate how to use Wireshark to capture and analyze network traffic exploited by attackers through the Telnet protocol. This project covers how to filter traffic for specific protocols, interpret hexadecimal views to identify sensitive information and utilize the TCP Stream feature to view the entire conversation in a readable format. By showcasing these techniques, the project highlights the security vulnerabilities of Telnet and emphasizes the importance of using more secure protocols like SSH.

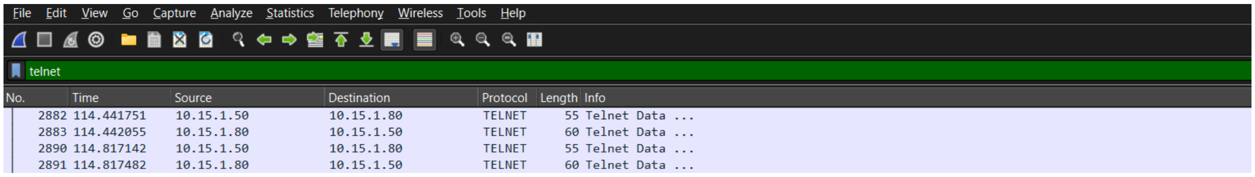
**Software and tools I use:** Wireshark.

**Project:** In this project, I will begin by capturing network traffic using Wireshark. After capturing traffic for a few minutes, I will stop the capture to analyze the network data, where I expect to find a variety of packets representing different network activities.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time      Source       Destination  Protocol Length Info
   1 0.000000  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
   2 0.000049  10.15.1.50   10.1.10.119  TCP       66 3389 → 51095 [ACK] Seq=1 Ack=4294967211 Win=63203 Len=0 SLE=4294967254 SRE=51
   3 0.016390  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
   4 0.016438  10.15.1.50   10.1.10.119  TCP       66 [TCP Dup ACK 2#1] 3389 → 51095 [ACK] Seq=1 Ack=4294967211 Win=63203 Len=0 SLE=4294967254 SRE=101
   5 0.016876  10.1.10.119  10.15.1.50   TCP      240 [TCP Retransmission] 51095 → 3389 [PSH, ACK] Seq=4294967211 Ack=1 Win=256 Len=186
   6 0.016928  10.15.1.50   10.1.10.119  TCP       66 3389 → 51095 [ACK] Seq=1 Ack=101 Win=63017 Len=0 SLE=4294967254 SRE=101
   7 0.030607  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
   8 0.046473  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
   9 0.046577  10.15.1.50   10.1.10.119  TCP       54 3389 → 51095 [ACK] Seq=1 Ack=201 Win=62917 Len=0
  10 0.057159  10.1.10.119  10.15.1.50   TLSv1.2   97 Application Data
  11 0.078296  10.1.10.119  10.15.1.50   TLSv1.2  111 Application Data
  12 0.078369  10.15.1.50   10.1.10.119  TCP       54 3389 → 51095 [ACK] Seq=1 Ack=301 Win=62817 Len=0
  13 0.094142  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
  14 0.110245  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
  15 0.110308  10.15.1.50   10.1.10.119  TCP       54 3389 → 51095 [ACK] Seq=1 Ack=401 Win=62717 Len=0
  16 0.124054  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
  17 0.140175  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
  18 0.140234  10.15.1.50   10.1.10.119  TCP       54 3389 → 51095 [ACK] Seq=1 Ack=501 Win=62617 Len=0
  19 0.158920  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
  20 0.172149  10.1.10.119  10.15.1.50   TLSv1.2  104 Application Data
```

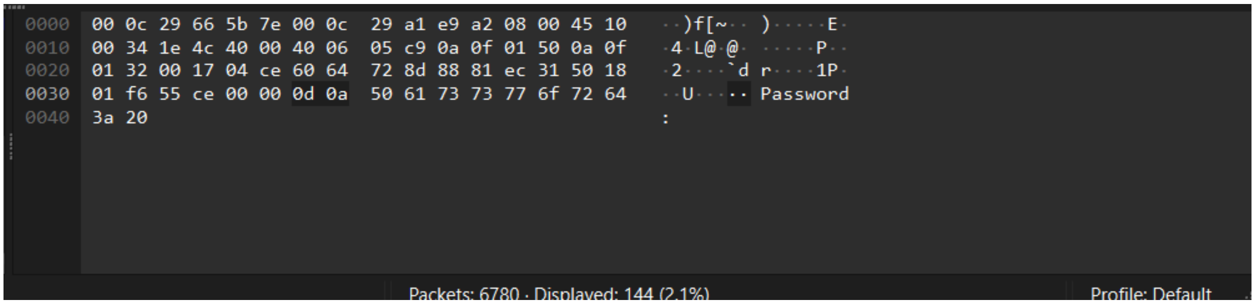
Wireshark offers a powerful feature that allows me to search for specific types of ports within the captured traffic. Knowing that the attacker exploited the Telnet port, I will use Wireshark's display filter to focus on Telnet traffic. This will help me analyze how the attacker used the Telnet port to gain unauthorized access.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
telnet

No.    Time        Source      Destination  Protocol Length Info
  2882 114.441751  10.15.1.50  10.15.1.80   TELNET    55 Telnet Data ...
  2883 114.442055  10.15.1.80  10.15.1.50   TELNET    60 Telnet Data ...
  2890 114.817142  10.15.1.50  10.15.1.80   TELNET    55 Telnet Data ...
  2891 114.817482  10.15.1.80  10.15.1.50   TELNET    60 Telnet Data ...
```

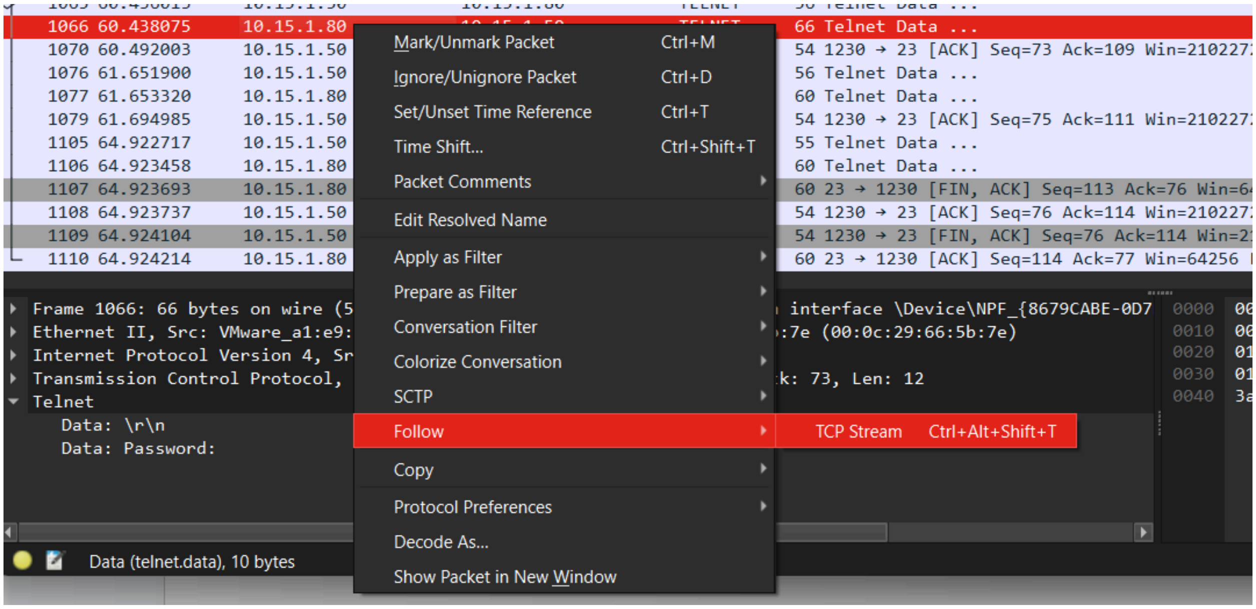
After filtering the traffic for Telnet protocols, I will review each Telnet session individually by examining the hexadecimal view in Wireshark. My goal is to identify any packets that may contain sensitive information, such as a password. Upon inspecting these sessions, I discovered one Telnet session where the password was displayed in the hexadecimal view. This confirms that in this Telnet session, the password was transmitted in plain text, highlighting the security risks associated with using Telnet.



```
0000  00 0c 29 66 5b 7e 00 0c  29 a1 e9 a2 08 00 45 10   ··)f[~·· )·····E·
0010  00 34 1e 4c 40 00 40 06  05 c9 0a 0f 01 50 0a 0f   ·4·L@·@· ·····P··
0020  01 32 00 17 04 ce 60 64  72 8d 88 81 ec 31 50 18   ·2····`d r····1P·
0030  01 f6 55 ce 00 00 0d 0a  50 61 73 73 77 6f 72 64   ··U····· Password
0040  3a 20                                              :
```

Packets: 6780 · Displayed: 144 (2.1%)    Profile: Default

After finding the Telnet session that transmitted the password in plain text, I will right-click on the selected packet and choose ‘Follow’ followed by ‘TCP Stream’. This action allows me to view the entire conversation between the client and server within that session. By following the TCP stream, I can observe the complete exchange of data, including any credentials or commands that were transmitted during the session, all displayed in an easily readable format.



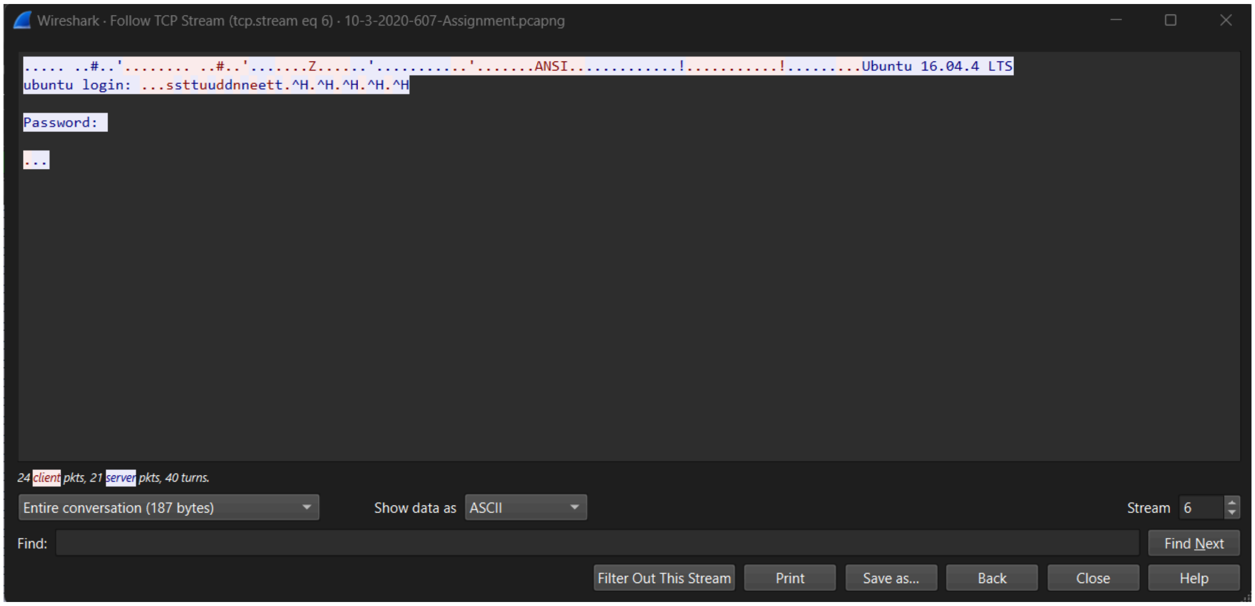
```
1066 60.438075  10.15.1.80   ...   66 Telnet Data ...
1070 60.492003  10.15.1.50   ...   54 1230 → 23 [ACK] Seq=73 Ack=109 Win=210227
1076 61.651900  10.15.1.50   ...   56 Telnet Data ...
1077 61.653320  10.15.1.80   ...   60 Telnet Data ...
1079 61.694985  10.15.1.50   ...   54 1230 → 23 [ACK] Seq=75 Ack=111 Win=210227
1105 64.922717  10.15.1.50   ...   55 Telnet Data ...
1106 64.923458  10.15.1.80   ...   60 Telnet Data ...
1107 64.923693  10.15.1.80   ...   60 23 → 1230 [FIN, ACK] Seq=113 Ack=76 Win=6
1108 64.923737  10.15.1.50   ...   54 1230 → 23 [ACK] Seq=76 Ack=114 Win=210227
1109 64.924104  10.15.1.50   ...   54 1230 → 23 [FIN, ACK] Seq=76 Ack=114 Win=2
1110 64.924214  10.15.1.80   ...   60 23 → 1230 [ACK] Seq=114 Ack=77 Win=64256

Frame 1066: 66 bytes on wire (5 ... interface \Device\NPF_{8679CABE-0D7
Ethernet II, Src: VMware_a1:e9: ... :7e (00:0c:29:66:5b:7e)
Internet Protocol Version 4, Sr
Transmission Control Protocol, ... k: 73, Len: 12
Telnet
    Data: \r\n
    Data: Password:

Data (telnet.data), 10 bytes
```

Mark/Unmark Packet Ctrl+M
Ignore/Unignore Packet Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift... Ctrl+Shift+T
Packet Comments
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
TCP Stream Ctrl+Alt+Shift+T
Copy
Protocol Preferences
Decode As...
Show Packet in New Window

It will show like that.



Wireshark · Follow TCP Stream (tcp.stream eq 6) · 10-3-2020-607-Assignment.pcapng

```
..... ..#..'........ ..#..'......Z......'...........'.......ANSI.............!...........!.........Ubuntu 16.04.4 LTS
ubuntu login: ...ssttuuddnneett.^H.^H.^H.^H.^H

Password: 

...
```

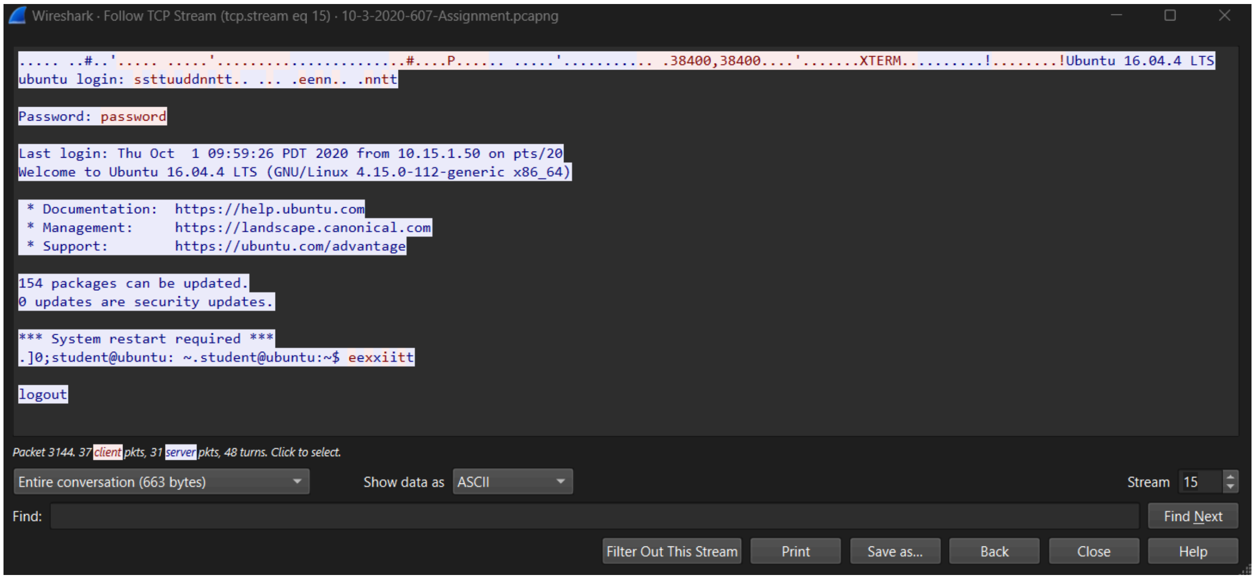
24 client pkts, 21 server pkts, 40 turns.

Entire conversation (187 bytes)

Show data as ASCII

Stream 6

My job is to keep changing Stream until I find the password. After doing that I found the password.



Wireshark · Follow TCP Stream (tcp.stream eq 15) · 10-3-2020-607-Assignment.pcapng

```
..... ..#..'..... .....'........................#....P...... .....'........... .38400,38400....'.......XTERM..........!........!Ubuntu 16.04.4 LTS
ubuntu login: ssttuuddnntt.. ... .eenn.. .nntt

Password: password

Last login: Thu Oct  1 09:59:26 PDT 2020 from 10.15.1.50 on pts/20
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-112-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

154 packages can be updated.
0 updates are security updates.

*** System restart required ***
.]0;student@ubuntu: ~.student@ubuntu:~$ eexxiitt

logout
```

Packet 3144. 37 client pkts, 31 server pkts, 48 turns. Click to select.

Entire conversation (663 bytes)

Show data as ASCII

Stream 15

This example highlights how Telnet exposes credentials in plain text, making it vulnerable to interception by attackers. Telnet always transmits passwords unencrypted and that's why it is not secure. In contrast, SSH (Secure Shell) encrypts passwords and other data, making it significantly harder for attackers to tamper with the information. Therefore, using SSH is a more secure alternative for protecting sensitive information during remote communications.