

Project-2: Comparative Analysis of Telnet and SSH Using Wireshark -

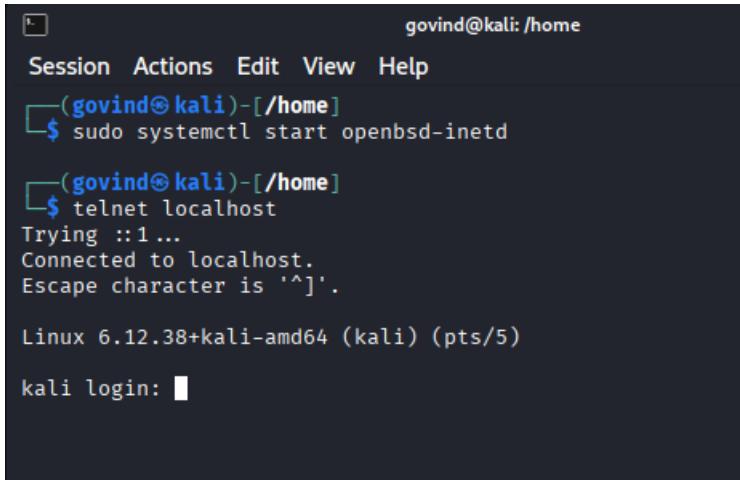
1. Kali Linux Hostname / User Configuration:

A new user (Govind) was created and user switching was tested to simulate real-world non-root access.

```
govind@kali: /home/kali
Session Actions Edit View Help
(kali㉿kali)-[~]
└─$ sudo adduser govind
[sudo] password for kali:
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for govind
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
(kali㉿kali)-[~]
└─$ sudo su govind
(govind㉿kali)-[/home/kali]
└─$ whoami
govind
(govind㉿kali)-[/home/kali]
└─$
```

2. Telnet Service Configuration and Activation:

Telnet service was enabled using openbsd-inetd to demonstrate insecure plaintext communication.



```
govind@kali: /home
Session Actions Edit View Help
└─(govind㉿kali)-[/home]
$ sudo systemctl start openbsd-inetd

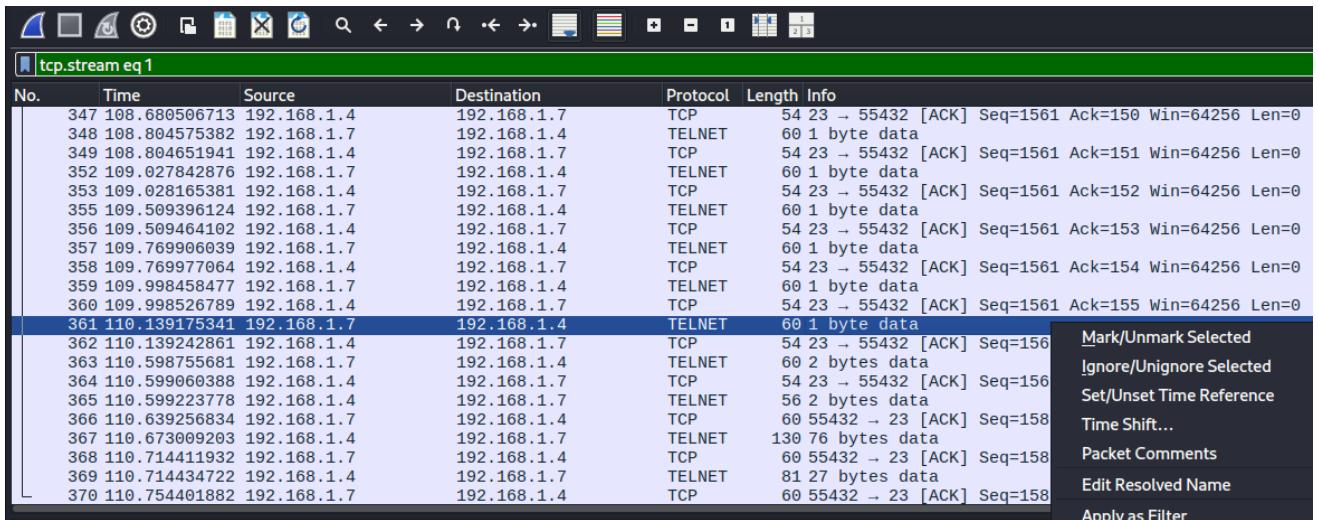
└─(govind㉿kali)-[/home]
$ telnet localhost
Trying ::1...
Connected to localhost.
Escape character is '^]'.

Linux 6.12.38+kali-amd64 (kali) (pts/5)

kali login: █
```

3. Wireshark Capturing Telnet Traffic:

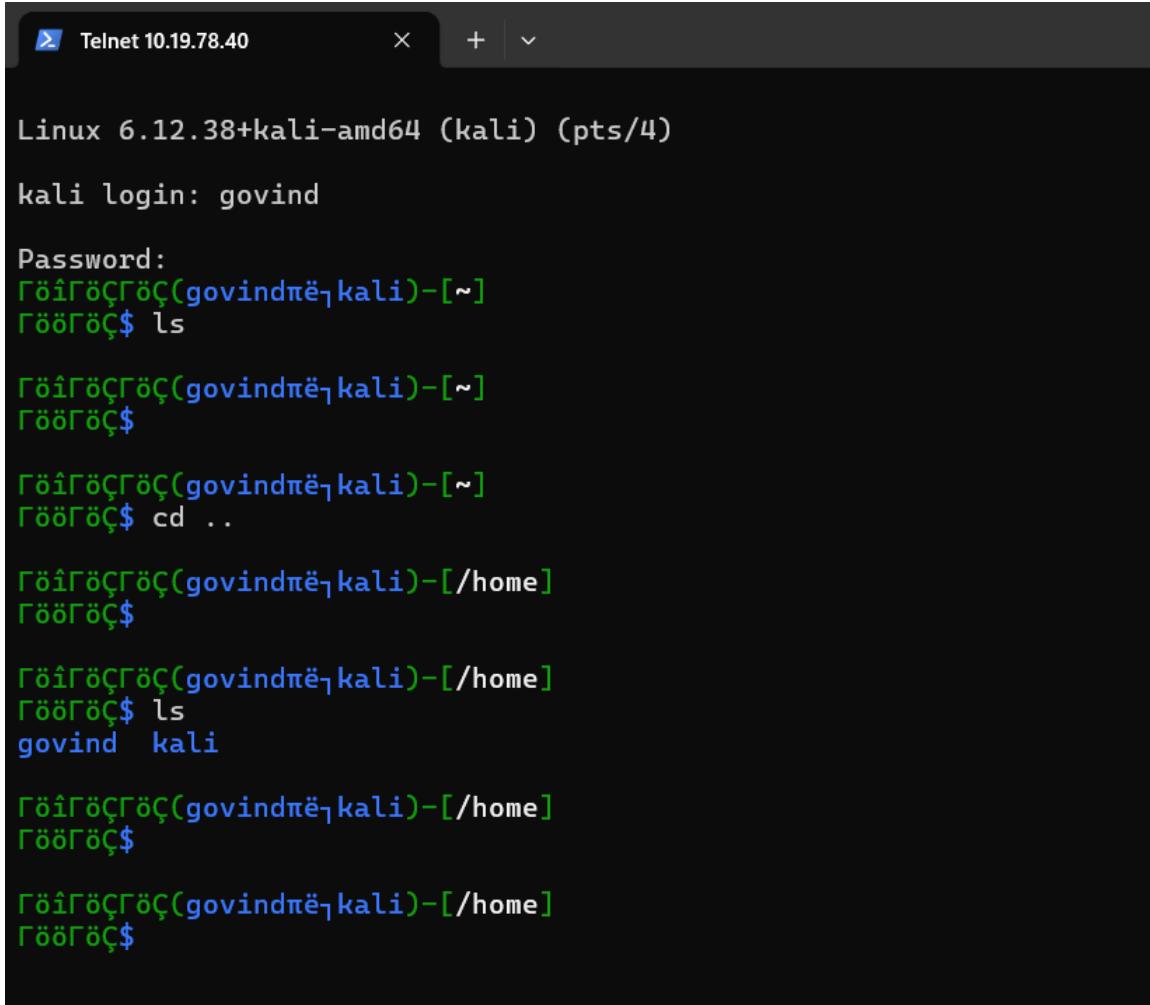
Wireshark was used on Kali Linux to capture Telnet packets on TCP port 23.



No.	Time	Source	Destination	Protocol	Length	Info
347	108.680506713	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=150 Win=64256 Len=0
348	108.804575382	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
349	108.804651941	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=151 Win=64256 Len=0
352	109.027842876	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
353	109.028165381	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=152 Win=64256 Len=0
355	109.509396124	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
356	109.509464102	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=153 Win=64256 Len=0
357	109.769906039	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
358	109.769977064	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=154 Win=64256 Len=0
359	109.998458477	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
360	109.998526789	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=1561 Ack=155 Win=64256 Len=0
361	110.139175341	192.168.1.7	192.168.1.4	TELNET	60	1 byte data
362	110.139242861	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=156
363	110.598755681	192.168.1.7	192.168.1.4	TELNET	60	2 bytes data
364	110.599060388	192.168.1.4	192.168.1.7	TCP	54	23 → 55432 [ACK] Seq=156
365	110.599223778	192.168.1.4	192.168.1.7	TELNET	56	2 bytes data
366	110.639256834	192.168.1.7	192.168.1.4	TCP	60	55432 → 23 [ACK] Seq=158
367	110.673009203	192.168.1.4	192.168.1.7	TELNET	130	76 bytes data
368	110.714411932	192.168.1.7	192.168.1.4	TCP	60	55432 → 23 [ACK] Seq=158
369	110.714434722	192.168.1.4	192.168.1.7	TELNET	81	27 bytes data
370	110.754401882	192.168.1.7	192.168.1.4	TCP	60	55432 → 23 [ACK] Seq=158

4. Telnet Login and Command Execution:

Telnet login was performed from Windows and commands were executed.



The screenshot shows a Telnet session window titled "Telnet 10.19.78.40". The session content is as follows:

```
Linux 6.12.38+kali-amd64 (kali) (pts/4)
kali login: govind
Password:
ΓöîΓöçΓöç(govindπë_1kali)-[~]
ΓööΓöç$ ls
ΓöîΓöçΓöç(govindπë_1kali)-[~]
ΓööΓöç$ 
ΓöîΓöçΓöç(govindπë_1kali)-[~]
ΓööΓöç$ cd ..
ΓöîΓöçΓöç(govindπë_1kali)-[/home]
ΓööΓöç$ 
ΓöîΓöçΓöç(govindπë_1kali)-[/home]
ΓööΓöç$ ls
govind kali
ΓöîΓöçΓöç(govindπë_1kali)-[/home]
ΓööΓöç$ 
ΓöîΓöçΓöç(govindπë_1kali)-[/home]
ΓööΓöç$
```

5. Plain-text Credentials Visible in Packets:

Telnet TCP stream shows username and password in plaintext.

Wireshark - Follow TCP Stream (tcp.stream eq 0) - eth0

..%..&..... .#.'..\$
..%..&..... .#.'..\$
.....!.....
....x.....!.....ANSI..
.....!".....!
.....!".....!
.....
.....

Linux 6.12.38+kali-amd64 (kali) (pts/4)

```
kali login:  
.....k  
k  
. ^H  
g  
o  
o  
v  
v  
i  
i  
n  
n  
d  
d  
. ^H  
. ^H  
. ^H
```

6. SSH Service Configuration and Activation:

SSH service was verified and started on Kali Linux.

```

govind@kali: ~
Session Actions Edit View Help
└─(kali㉿kali)-[~]
└─$ sudo su govind
[sudo] password for kali:
└─(govind㉿kali)-[/home/kali]
└─$ sudo systemctl start ssh
[sudo] password for govind:

└─(govind㉿kali)-[/home/kali]
└─$ ssh localhost
govind@localhost's password:
Linux kali 6.12.38+kali1-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.38-1kali1 (202
5-08-12) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Dec 25 03:20:35 2025 from ::1
└─(govind㉿kali)-[~]
└─$
```

7. Wireshark Capturing SSH Traffic:

Wireshark captured SSH traffic on TCP port 22.

No.	Time	Source	Destination	Protocol	Length	Info
17	40.441976821	192.168.1.7	192.168.1.4	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_9.5)
19	40.450356727	192.168.1.4	192.168.1.7	SSHv2	87	Server: Protocol (SSH-2.0-OpenSSH_10.2p1 Debian-3)
20	40.459104193	192.168.1.4	192.168.1.7	SSHv2	1094	Server: Key Exchange Init
22	40.466988745	192.168.1.7	192.168.1.4	SSHv2	1486	Client: Key Exchange Init
24	40.508767061	192.168.1.7	192.168.1.4	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
26	40.512470544	192.168.1.4	192.168.1.7	SSHv2	546	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys,
27	40.516487848	192.168.1.7	192.168.1.4	SSHv2	70	Client: New Keys
31	40.557678975	192.168.1.7	192.168.1.4	SSHv2	98	Client: Encrypted packet (len=44)
33	40.557875460	192.168.1.4	192.168.1.7	SSHv2	98	Server: Encrypted packet (len=44)
34	40.558318225	192.168.1.7	192.168.1.4	SSHv2	122	Client: Encrypted packet (len=68)
35	40.561657766	192.168.1.4	192.168.1.7	SSHv2	106	Server: Encrypted packet (len=52)
42	51.278751164	192.168.1.7	192.168.1.4	SSHv2	202	Client: Encrypted packet (len=148)
45	53.396968954	192.168.1.4	192.168.1.7	SSHv2	106	Server: Encrypted packet (len=52)
67	128.225295854	192.168.1.7	192.168.1.4	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_9.5)
69	128.233289002	192.168.1.4	192.168.1.7	SSHv2	87	Server: Protocol (SSH-2.0-OpenSSH_10.2p1 Debian-3)
70	128.236554932	192.168.1.7	192.168.1.4	SSHv2	1486	Client: Key Exchange Init
71	128.241848773	192.168.1.4	192.168.1.7	SSHv2	1094	Server: Key Exchange Init
72	128.243368667	192.168.1.7	192.168.1.4	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
73	128.247259023	192.168.1.4	192.168.1.7	SSHv2	546	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys,
74	128.251282195	192.168.1.7	192.168.1.4	SSHv2	70	Client: New Keys
76	128.292729784	192.168.1.7	192.168.1.4	SSHv2	98	Client: Encrypted packet (len=44)
78	128.293060623	192.168.1.4	192.168.1.7	SSHv2	98	Server: Encrypted packet (len=44)
79	128.293350724	192.168.1.7	192.168.1.4	SSHv2	122	Client: Encrypted packet (len=68)
80	128.294508691	192.168.1.4	192.168.1.7	SSHv2	106	Server: Encrypted packet (len=52)
83	132.533014028	192.168.1.7	192.168.1.4	SSHv2	202	Client: Encrypted packet (len=148)
85	132.612166408	192.168.1.4	192.168.1.7	SSHv2	82	Server: Encrypted packet (len=52)
86	132.617637815	192.168.1.7	192.168.1.4	SSHv2	166	Client: Encrypted packet (len=148)
88	132.661448887	192.168.1.4	192.168.1.7	SSHv2	682	Server: Encrypted packet (len=148)
89	132.662248800	192.168.1.4	192.168.1.7	SSHv2	98	Client: Encrypted packet (len=68)
91	132.665195477	192.168.1.7	192.168.1.4	SSHv2	106	Server: Encrypted packet (len=52)
93	132.666172179	192.168.1.4	192.168.1.7	SSHv2	126	Client: Encrypted packet (len=148)
94	132.669987799	192.168.1.4	192.168.1.7	SSHv2	130	Server: Encrypted packet (len=52)
95	132.669987826	192.168.1.4	192.168.1.7	SSHv2	158	Client: Encrypted packet (len=148)
97	132.670298771	192.168.1.4	192.168.1.7	SSHv2	126	Server: Encrypted packet (len=68)

Context menu options visible in the screenshot:

- Mark/Unmark Selected
- Ignore/Unignore Selected
- Set/Unset Time Reference
- Time Shift...
- Packet Comments
- Edit Resolved Name
- Apply as Filter
- Prepare as Filter
- Conversation Filter
- Colorize Conversation
- SCTP
- Follow
- TCP Stream

8. SSH Login and Command Execution:

SSH login from Windows to Kali was successful.

```

govind@kali: /home
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadLine'.

PS C:\Users\ASUS> ssh govind@10.19.78.40
govind@10.19.78.40's password:
Linux kali 6.12.38+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.38-1kali1 (2025-08-12) x86_64

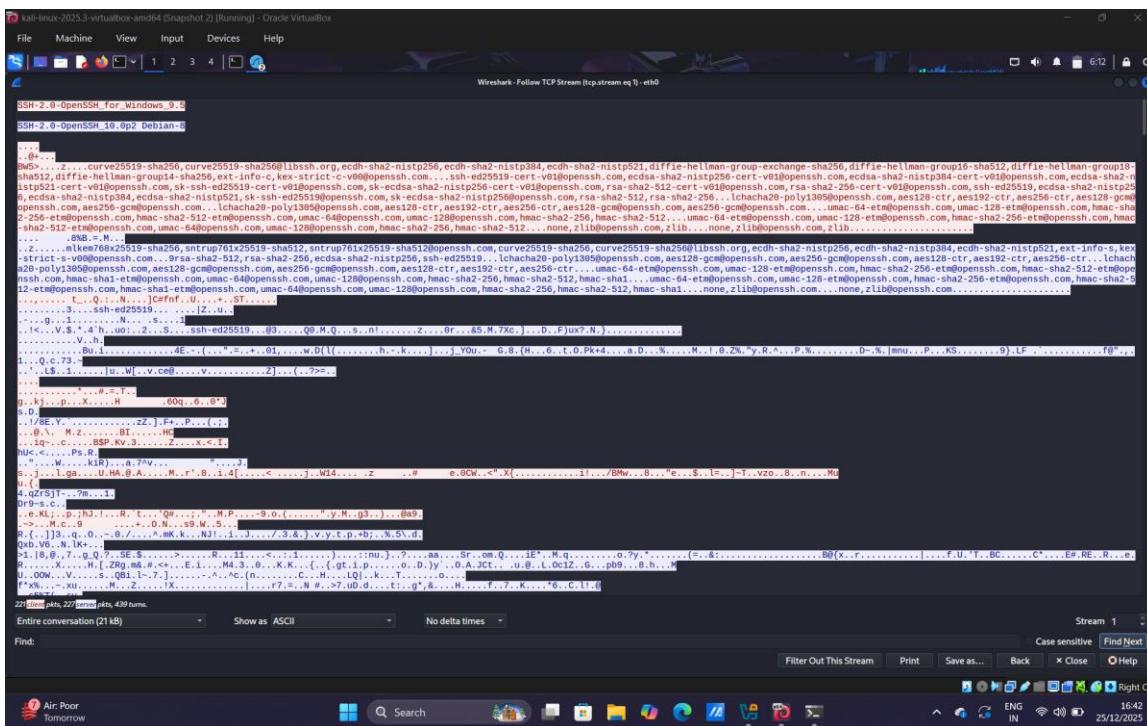
The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Dec 25 06:06:01 2025 from ::1
[Govind@kali ~]
$ ls
[Govind@kali ~]
$ cd ..
[Govind@kali ~]
$ 

```

9. Encrypted SSH Packets (Unreadable Payload)

SSH TCP stream shows encrypted unreadable payload.



Final Comparison Summary

Telnet transmits credentials in plaintext and is insecure, whereas SSH encrypts all communication, making it safe against packet sniffing attacks.

Telnet runs over TCP port 23 and does not encrypt data. Enabling it intentionally creates an insecure service to demonstrate real-world risks in legacy systems.

Wireshark captures packets at the network interface level. By filtering Telnet traffic, we can clearly observe how data travels without encryption.

User credentials and commands are transmitted directly over the network. This simulates how attackers can sniff sensitive data on unsecured networks.

The visibility of usernames and passwords proves Telnet is insecure. This is a critical vulnerability that violates basic security principles.

SSH replaces Telnet by providing encrypted remote access. It is the industry standard for secure system administration.

Although packets are captured, their content is unreadable. This demonstrates the effectiveness of cryptographic protection.

Even login credentials and commands remain encrypted. This protects users against packet sniffing and man-in-the-middle attacks.

The payload appears as random data due to encryption algorithms. This ensures confidentiality even if traffic is intercepted.

Telnet should never be used in modern networks. SSH is secure, reliable, and resistant to passive network attacks.