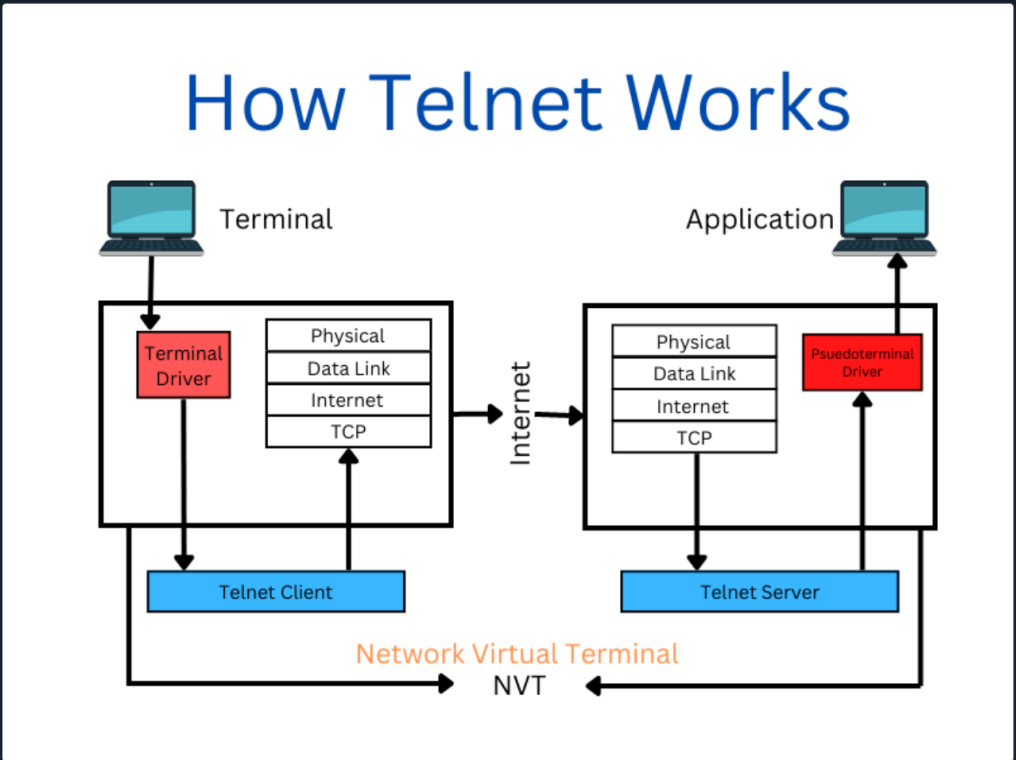
Strange Telnet & UDP Connections

When we look for strange traffic, we should always consider telnet and UDP traffic. After all, these can be overlooked, but can especially revealing during our traffic analysis efforts.

Telnet



Telnet is a network protocol that allows a bidirectional interactive communication session between two devices over a network. This protocol was developed in the 1970s and was defined in RFC 854. As of recent years, its usage has decreased significantly as opposed to SSH.

In many older cases, such as our Windows NT like machines, they may still utilize telnet to provide remote command and control to microsoft terminal services.

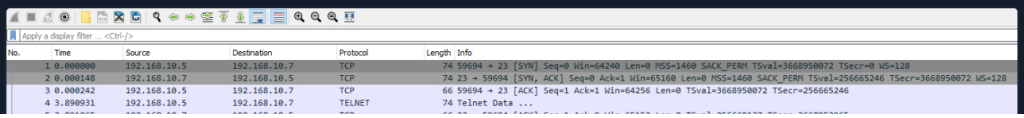
However, we should always watch for weird and strange telnet communications as it can also be used by attackers for malicious purposes such as data exfiltration and tunneling.

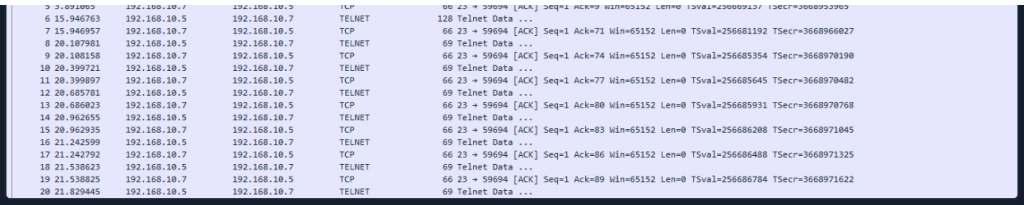
Finding Traditional Telnet Traffic Port 23

Related PCAP File(s):

- `telnet_tunneling_23.pcapng`

Suppose we were to open Wireshark, we might notice some telnet communications originating from Port 23. In this case, we can always inspect this traffic further.





No.	Time	Source	Destination	Protocol	Length	Info
6	15.946763	192.168.10.5	192.168.10.7	TELNET	128	Telnet Data ...
7	15.946957	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=71 Win=65152 Len=0 TSval=256681192 TSecr=3668966027
8	20.107981	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
9	20.108158	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=74 Win=65152 Len=0 TSval=256685354 TSecr=3668970190
10	20.399721	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
11	20.399897	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=77 Win=65152 Len=0 TSval=256685645 TSecr=3668970482
12	20.685781	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
13	20.686023	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=80 Win=65152 Len=0 TSval=256685931 TSecr=3668970768
14	20.962655	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
15	20.962935	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=83 Win=65152 Len=0 TSval=256686208 TSecr=3668971045
16	21.242599	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
17	21.242792	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=86 Win=65152 Len=0 TSval=256686488 TSecr=3668971325
18	21.538623	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...
19	21.538825	192.168.10.7	192.168.10.5	TCP	66	23 → 59694 [ACK] Seq=1 Ack=89 Win=65152 Len=0 TSval=256686784 TSecr=3668971622
20	21.829445	192.168.10.5	192.168.10.7	TELNET	69	Telnet Data ...

Fortunately for us, telnet traffic tends to be decrypted and easily inspectable, but like ICMP, DNS, and other tunneling methods, attackers may encrypt, encode, or obfuscate this text. So we should always be careful.

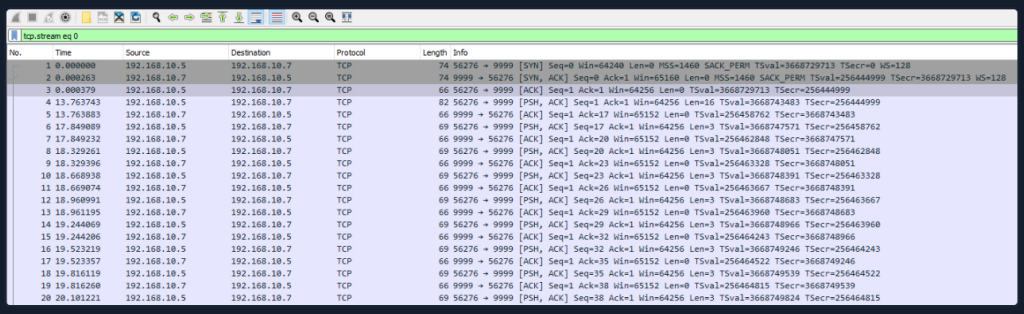
```
> Frame 6: 128 bytes on wire (1024 bits), 128 bytes captured (1024 bits) on interface \Device\NPF_{CCC4B960-1E92-4BD5-BBF3-11E2DFD12FE1}, id 0
> Ethernet II, Src: PcsCompu_53:0c:ba (08:00:27:53:0c:ba), Dst: Micro-St_95:68:2a (44:8a:5b:95:68:2a)
> Internet Protocol Version 4, Src: 192.168.10.5, Dst: 192.168.10.7
> Transmission Control Protocol, Src Port: 59694, Dst Port: 23, Seq: 9, Ack: 1, Len: 62
v Telnet
    Data: telnet is unencrypted, so we can find things a little easier\r\n
```

Unrecognized TCP Telnet in Wireshark

Related PCAP File(s):

- telnet_tunneling_9999.pcapng

Telnet is just a communication protocol, and as such can be easily switched to another port by an attacker. Keeping an eye on these strange port communications can allow us to find potentially malicious actions. Lets take the following for instance.

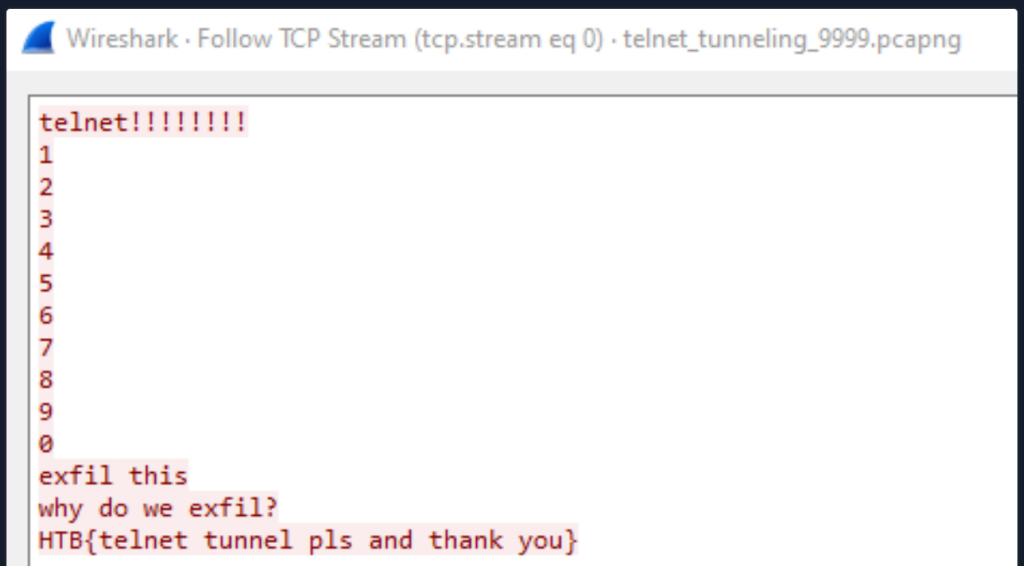


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.10.5	192.168.10.7	TCP	74	56276 → 9999 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3668729713 TSecr=0 WS=128
2	0.000263	192.168.10.7	192.168.10.5	TCP	74	9999 → 56276 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=256444999 TSecr=3668729713 WS=128
3	0.000379	192.168.10.5	192.168.10.7	TCP	66	56276 → 9999 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3668729713 TSecr=256444999
4	13.763743	192.168.10.5	192.168.10.7	TCP	82	56276 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=16 TSval=3668743483 TSecr=256444999
5	13.763883	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=17 Win=65152 Len=0 TSval=256458762 TSecr=3668743483
6	17.849089	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=17 Ack=1 Win=64256 Len=3 TSval=3668747571 TSecr=256458762
7	17.849232	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=20 Win=65152 Len=0 TSval=256462848 TSecr=3668747571
8	18.329261	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=20 Ack=1 Win=64256 Len=3 TSval=3668748051 TSecr=256462848
9	18.329396	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=23 Win=65152 Len=0 TSval=256463328 TSecr=3668748051
10	18.668938	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=23 Ack=1 Win=64256 Len=3 TSval=3668748391 TSecr=256463328
11	18.669074	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=26 Win=65152 Len=0 TSval=256463667 TSecr=3668748391
12	18.960991	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=26 Ack=1 Win=64256 Len=3 TSval=3668748683 TSecr=256463667
13	18.961195	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=29 Win=65152 Len=0 TSval=256463960 TSecr=3668748683
14	19.244069	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=29 Ack=1 Win=64256 Len=3 TSval=3668748966 TSecr=256463960
15	19.244206	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=32 Win=65152 Len=0 TSval=256464243 TSecr=3668748966
16	19.523219	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=32 Ack=1 Win=64256 Len=3 TSval=3668749246 TSecr=256464243
17	19.523357	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=35 Win=65152 Len=0 TSval=256464522 TSecr=3668749246
18	19.816119	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=35 Ack=1 Win=64256 Len=3 TSval=3668749539 TSecr=256464522
19	19.816260	192.168.10.7	192.168.10.5	TCP	66	9999 → 56276 [ACK] Seq=1 Ack=38 Win=65152 Len=0 TSval=256464815 TSecr=3668749539
20	20.101221	192.168.10.5	192.168.10.7	TCP	69	56276 → 9999 [PSH, ACK] Seq=38 Ack=1 Win=64256 Len=3 TSval=3668749824 TSecr=256464815

We may see a ton of communications from one client on port 9999. We can dive into this a little further by looking at the contents of these communications.

```
0000  44 8a 5b 95 68 2a 08 00  27 53 0c ba 08 00 45 00   D·[·h*·· 'S····E·
0010  00 44 04 a4 40 00 40 06  a0 b3 c0 a8 0a 05 c0 a8   ·D··@·@· ········
0020  0a 07 db d4 27 0f 00 d8  74 8c e8 e9 3c f9 80 18   ····'··· t···<···
0030  01 f6 df d7 00 00 01 01  08 0a da ac 95 3b 0f 49   ········ ·····;·I
0040  0a 47 74 65 6c 6e 65 74  21 21 21 21 21 21 21 21   ·Gtelnet !!!!!!!!
0050  0d 0a                                             ··
```

If we noticed something like above, we would want to follow this TCP stream.



OFFLINE

Start Instance

∞ / 1 spawns left

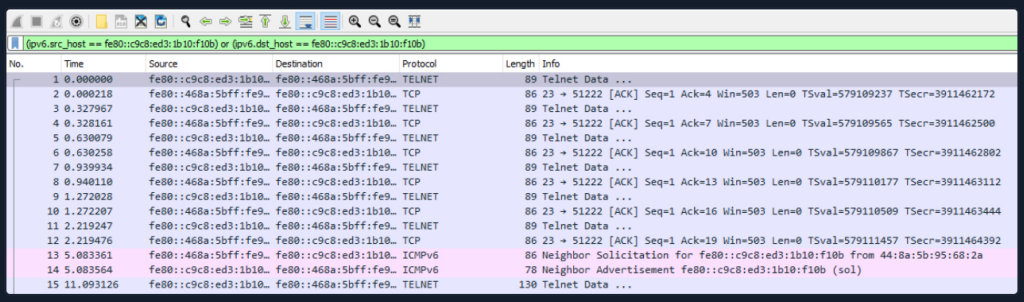
Doing so can allow us to inspect potentially malicious actions.

Telnet Protocol through IPv6

Related PCAP File(s):

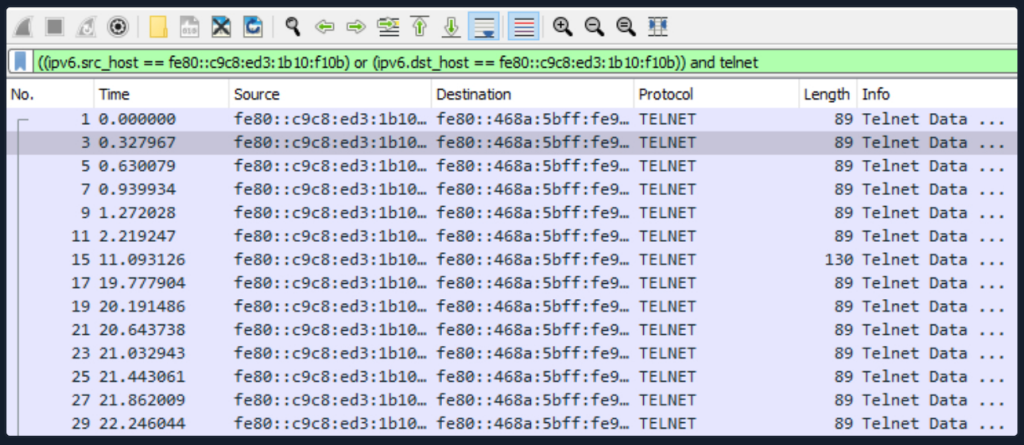
- `telnet_tunneling_ipv6.pcapng`

After all, unless our local network is configured to utilize IPv6, observing IPv6 traffic can be an indicator of bad actions within our environment. We might notice the usage of IPv6 addresses for telnet like the following.

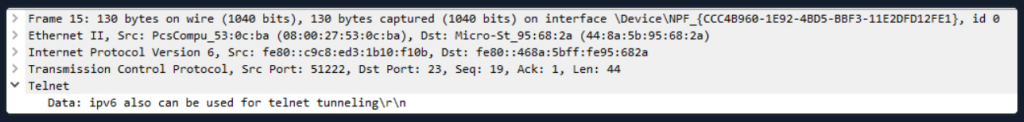


We can narrow down our filter in Wireshark to only show telnet traffic from these addresses with the following filter.

- `((ipv6.src_host == fe80::c9c8:ed3:1b10:f10b) or (ipv6.dst_host == fe80::c9c8:ed3:1b10:f10b)) and telnet`



Likewise, we can inspect the contents of these packets through their data field, or by following the TCP stream.

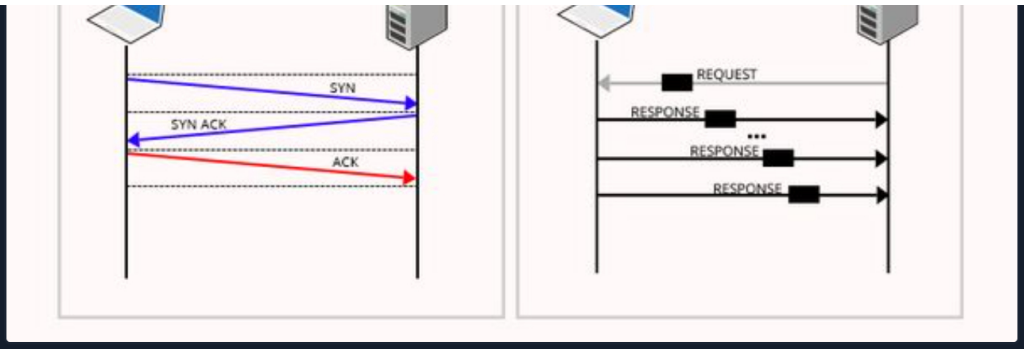


Watching UDP Communications

Related PCAP File(s):

- `udp_tunneling.pcapng`

On the other hand, attackers might opt to use UDP connections over TCP in their exfiltration efforts.



One of the biggest distinguishing aspects between TCP and UDP is that UDP is connectionless and provides fast transmission. Let's take the following traffic for instance.

udp

No.	Time	Source	Destination	Protocol	Length	Info
15	10.178286	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
16	10.864737	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
17	11.595531	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
18	12.082710	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
19	12.706286	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
20	13.584752	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
21	13.997879	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
22	14.703007	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
23	21.214295	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
24	21.819600	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
25	22.826006	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
26	24.276725	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
27	26.929886	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
28	27.328421	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
29	27.692619	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
30	28.033756	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
31	28.372636	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2
32	28.695390	192.168.10.5	192.168.10.7	UDP	60	46678 → 12345 Len=2

We will notice that instead of a SYN, SYN/ACK, ACK sequence, the communications are immediately sent over to the recipient. Like TCP, we can follow UDP traffic in Wireshark, and inspect its contents.

Wireshark · Follow UDP Stream (udp.stream eq 0) · udp_tunneling.pcapng

```
abcdefghijklmnopqrstuvwxyz12345678
```

```
9
10
UDP can be superb for exfil
especially in the case of quick connections, unlike TCP, which is super monitored
Hence why we attackers like it
```

Common Uses of UDP

UDP although less reliable than TCP provides quicker connections through its connectionless state. As such, we might find legitimate traffic that uses UDP like the following:

Step	Description
1. Real-time Applications	Applications like streaming media, online gaming, real-time voice and video communications
2. DNS (Domain Name System)	DNS queries and responses use UDP
3. DHCP (Dynamic Host Configuration Protocol)	DHCP uses UDP to assign IP addresses and configuration information to network devices.
4. SNMP (Simple Network Management Protocol)	SNMP uses UDP for network monitoring and management
5. TFTP (Trivial File Transfer Protocol)	TFTP uses UDP for simple file transfers, commonly used by older Windows systems and others.

Answer the question(s) below to complete this Section and earn cubes!

+ 1 Inspect the telnet_tunneling_ipv6.pcapng file, part of this module's resources, and enter the hidden flag as your answer. Answer format: HTB(___) (Replace all spaces with underscores)

HTB(Ipv6_is_my_best_friend)

Submit

Previous Next

Mark Complete & Next

Powered by HACKTHEBOX