Wireshark and Netcat

Wire shark is a network protocol analyzer tool that demonstrates what's occurring on your network at a microscopic level. It's useful for determining network problems, analyzing security or potential vulnerabilities and debug protocol implementations. It's also useful for educational reasons to learn the inside of network protocols and the internal detail.

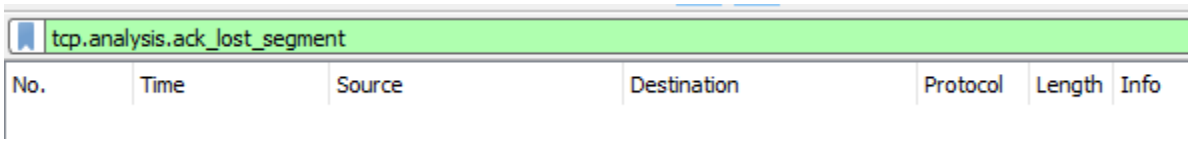
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Netgear_13:4d:b5	Spanning-tree-(for-…	STP	60	Conf. Root = 32768/0/9c:c9:eb:13:4d:b5 Cost = 0 Port = 0x8001
2	0.469960	Ubiquiti_07:9d:71	Broadcast	ARP	60	Who has 192.168.1.105? Tell 192.168.1.1
3	0.740234	192.168.1.123	74.125.21.189	UDP	75	60016 → 443 Len=33
4	0.747656	74.125.21.189	192.168.1.123	UDP	67	443 → 60016 Len=25
5	1.020866	192.168.1.123	199.232.34.248	TCP	55	10598 → 443 [ACK] Seq=1 Ack=1 Win=1025 Len=1 [TCP segment of a reassembled PDU]
6	1.025448	199.232.34.248	192.168.1.123	TCP	66	443 → 10598 [ACK] Seq=1 Ack=2 Win=133 Len=0 SLE=1 SRE=2
7	1.036251	192.168.1.123	199.232.34.248	TCP	55	10599 → 443 [ACK] Seq=1 Ack=1 Win=1025 Len=1 [TCP segment of a reassembled PDU]
8	1.040730	199.232.34.248	192.168.1.123	TCP	66	443 → 10599 [ACK] Seq=1 Ack=2 Win=133 Len=0 SLE=1 SRE=2
9	1.496527	Ubiquiti_07:9d:71	Broadcast	ARP	60	Who has 192.168.1.105? Tell 192.168.1.1
10	1.764843	192.168.1.123	64.233.177.83	TCP	1484	10503 → 443 [ACK] Seq=1 Ack=1 Win=1023 Len=1430 [TCP segment of a reassembled PDU]
11	1.764843	192.168.1.123	64.233.177.83	TLSv1.2	1213	Application Data
12	1.764869	192.168.1.123	64.233.177.83	TLSv1.2	162	Application Data
13	1.770456	64.233.177.83	192.168.1.123	TCP	60	443 → 10503 [ACK] Seq=1 Ack=1431 Win=473 Len=0
14	1.770485	64.233.177.83	192.168.1.123	TCP	60	443 → 10503 [ACK] Seq=1 Ack=2590 Win=484 Len=0
15	1.770533	64.233.177.83	192.168.1.123	TCP	60	443 → 10503 [ACK] Seq=1 Ack=2698 Win=484 Len=0
16	1.831220	192.168.1.123	192.168.1.175	TCP	164	10247 → 8009 [PSH, ACK] Seq=1 Ack=1 Win=1024 Len=110 [TCP segment of a reassembled PD
17	1.836042	192.168.1.175	192.168.1.123	TCP	164	8009 → 10247 [PSH, ACK] Seq=1 Ack=111 Win=430 Len=110 [TCP segment of a reassembled P
18	1.863268	64.233.177.83	192.168.1.123	TLSv1.2	442	Application Data
19	1.863386	64.233.177.83	192.168.1.123	TLSv1.2	795	Application Data
20	1.863401	192.168.1.123	64.233.177.83	TCP	54	10503 → 443 [ACK] Seq=2698 Ack=1130 Win=1027 Len=0
21	1.863651	64.233.177.83	192.168.1.123	TLSv1.2	164	Application Data
22	1.865923	64.233.177.83	192.168.1.123	TLSv1.2	141	Application Data
23	1.865942	192.168.1.123	64.233.177.83	TCP	54	10503 → 443 [ACK] Seq=2698 Ack=1327 Win=1027 Len=0
24	1.866072	64.233.177.83	192.168.1.123	TLSv1.2	93	Application Data
25	1.866194	192.168.1.123	64.233.177.83	TLSv1.2	93	Application Data
26	1.871454	64.233.177.83	192.168.1.123	TCP	60	443 → 10503 [ACK] Seq=1366 Ack=2737 Win=484 Len=0
27	1.878002	192.168.1.123	192.168.1.175	TCP	54	10247 → 8009 [ACK] Seq=111 Ack=111 Win=1023 Len=0
28	1.940148	192.168.1.123	64.233.177.189	UDP	75	53516 → 443 Len=33
29	1.947519	64.233.177.189	192.168.1.123	UDP	67	443 → 53516 Len=25
30	1.999988	Netgear_13:4d:b5	Spanning-tree-(for-…	STP	60	Conf. Root = 32768/0/9c:c9:eb:13:4d:b5 Cost = 0 Port = 0x8001
31	2.517293	192.168.1.123	192.168.1.175	TCP	164	10528 → 8009 [PSH, ACK] Seq=1 Ack=1 Win=1024 Len=110 [TCP segment of a reassembled PD

This is a screen shot of the network activity going through and from my computer. My computer is plugged in via an ethernet cable.

```
> Internet Protocol Version 4, Src: 192.168.1.123, Dst: 192.168.1.175
v Transmission Control Protocol, Src Port: 10247, Dst Port: 8009, Seq: 1, Ack: 1, Len: 110
    Source Port: 10247
    Destination Port: 8009
    [Stream index: 1]
    [TCP Segment Len: 110]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1804726266
    [Next Sequence Number: 111    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 2583031367
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
    Window: 1024
    [Calculated window size: 1024]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x8503 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (110 bytes)
    TCP segment data (110 bytes)
```

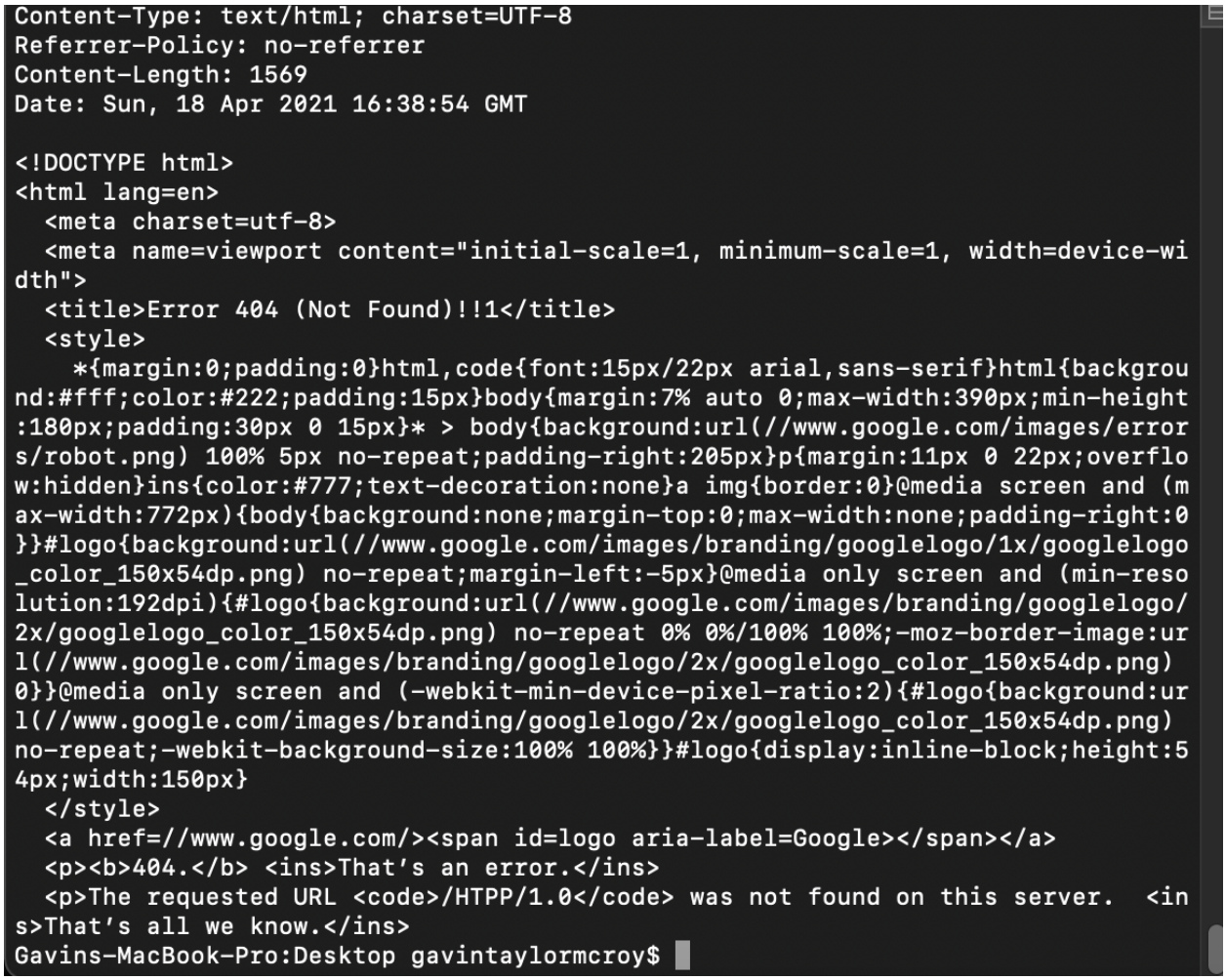
```
0000  00 f6 20 6f 7d 0d 44 8a  5b 9c bf 86 08 00 45 00   ·· o}·D· [·····E·
0010  00 96 60 75 40 00 80 06  00 00 c0 a8 01 7b c0 a8   ··`u@··· ·····{··
0020  01 af 28 07 1f 49 6b 91  ef fa 99 f5 ee 47 50 18   ··(··Ik· ·····GP·
0030  04 00 85 03 00 00 17 03  03 00 69 bd 35 81 ff d7   ········ ··i·5···
0040  36 fd 97 be 63 30 5c 3d  18 61 b0 3c 49 1e 77 1e   6···c0\= ·a·<I·w·
0050  58 f5 f0 21 a0 32 a7 81  1b 0a 19 16 48 c6 f8 18   X··!·2·· ····H···
0060  c3 92 c5 42 1f a0 65 fd  ef 87 f6 c2 f5 9b 3f af   ···B··e· ······?·
0070  6d 38 41 22 49 14 3b 95  15 c1 4c 77 90 9f e1 e6   m8A"I·;· ··Lw····
0080  22 4e 4d d1 9f ee 25 a4  07 71 93 2d 0b 42 e2 0a   "NM···%· ·q·-·B··
0090  1e ac 1d 1d 46 41 60 c9  91 2b 96 38 4b 9f e3 f7   ····FA`· ·+·8K···
00a0  70 ba 12 15                                       p···
```

Analyzing a TCP packet sent from my computer to google. Packet is encrypted



Wire shark also allows you to filter out specific packets and to find out if there is network loss or if a particular packet was re-sent. There are a ton of different filters from a particular source destination up to which protocol is being used.

Netcat - is a computer networking tool used for reading and writing to network connections using either TCP or UDP. The tool was designed to act as a dependable back end that can either be used by or driven by other programs and scripts. Also allows for port scanning and port listening



You can also request HTML pages such as google's home screen. You can do this with the following command.

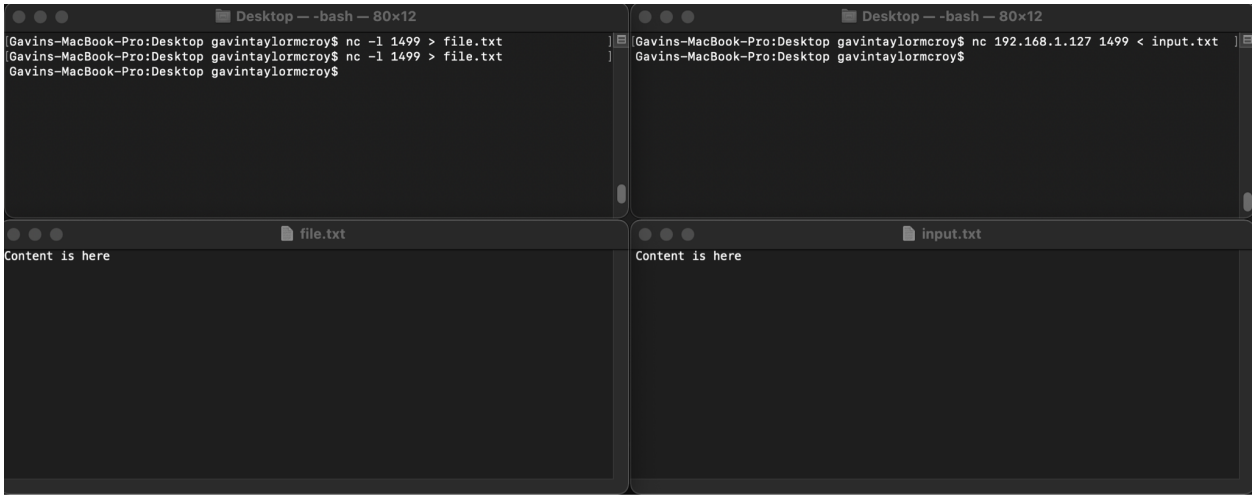
```
printf "GET /HTTP/1.0\r\n\t\n" | nc google.com 80;
```

You can also transfer files over netcat. Run this command on the server instance

```
nc -l 1499 > file.out
```

Then this command on the client instance

```
nc server.name 1499 < fileName.in
```



Input was the file I passed to file.txt
The content inside input was written into file.txt

As mentioned netcat can also listen in on ports. You can do this with this following command

nc -l 25

It will listen in on the port and notify via the terminal of any activity