

# Assignment 3

---

- Name: Abhiroop Mukherjee
- Roll No.: 510519109
- GSuite: [510519109.abhirup@students.iests.ac.in](mailto:510519109.abhirup@students.iests.ac.in)
- Subject: Computer Networks Lab (CS 3272)

## Question 1

---

Analyse the packets (across all layers) exchanged with your computer while executing the following commands:

1. ping
2. traceroute
3. dig
4. arp
5. wget



## Echo Reply Packet

No.	Time	Source	Destination	Protocol	Length	Info
22	1.373782	192.168.0.115	172.217.167.164	ICMP	74	Echo (ping) request id=0x0001, seq=255/65280, ttl=128 (reply in 29)
29	1.467644	172.217.167.164	192.168.0.115	ICMP	74	Echo (ping) reply id=0x0001, seq=255/65280, ttl=116 (request in 22)
36	2.395736	192.168.0.115	172.217.167.164	ICMP	74	Echo (ping) request id=0x0001, seq=256/1, ttl=128 (reply in 37)
37	2.407289	172.217.167.164	192.168.0.115	ICMP	74	Echo (ping) reply id=0x0001, seq=256/1, ttl=116 (request in 36)
45	3.409669	192.168.0.115	172.217.167.164	ICMP	74	Echo (ping) request id=0x0001, seq=257/257, ttl=128 (reply in 46)
46	3.422527	172.217.167.164	192.168.0.115	ICMP	74	Echo (ping) reply id=0x0001, seq=257/257, ttl=116 (request in 45)
50	4.433951	192.168.0.115	172.217.167.164	ICMP	74	Echo (ping) request id=0x0001, seq=258/513, ttl=128 (reply in 51)
51	4.445350	172.217.167.164	192.168.0.115	ICMP	74	Echo (ping) reply id=0x0001, seq=258/513, ttl=116 (request in 50)

```

> Frame 29: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{A6D5012E-CD41-4702-B6F5-74CE307F17F8}, id 0
> Ethernet II, Src: TendaTec_09:95:c0 (50:2b:73:09:95:c0), Dst: IntelCor_a0:4e:11 (a4:c3:f0:a0:4e:11)
└ Internet Protocol Version 4, Src: 172.217.167.164, Dst: 192.168.0.115
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x0000 (0)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 116
  Protocol: ICMP (1) TTL and Protocol
  Header Checksum: 0x3128 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.217.167.164 Source and Destination IP
  Destination Address: 192.168.0.115
  < Internet Control Message Protocol
    Type: 0 (Echo (ping) reply) Request Type
    Code: 0
    Checksum: 0x545c [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence Number (BE): 255 (0x00ff)
    Sequence Number (LE): 65280 (0xffff)
    [Request frame: 22]
    [Response time: 93.862 ms]
    Data (32 bytes)
  0000 a4 c3 f0 a0 4e 11 50 2b 73 09 95 c0 08 00 45 00 .....N P+ s.....E
  0010 00 3c 00 00 00 00 74 01 31 28 ac d9 a7 a4 c0 a8 <....t 1(.....
  0020 00 73 00 00 54 5c 00 01 00 ff 61 62 63 64 65 66 .s..T\... abcdef
  0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
  0040 77 61 62 63 64 65 66 67 68 69 wabcdef h1

```

- Protocol: ICMP
- Type: 0 (Echo (ping) reply)
- TTL: 116

## Protocol Info

Layer	Protocol Used
Network	ICMP

NOTE: ICMP is not associated with a transport layer protocol like TCP or UDP.

## 2. traceroute

The way **traceroute** works is by sending packets with increasing TTL and rely on "TTL Exceeded" packets from the routers to trace the path to a host.

Running **traceroute -I www.google.com** in terminal and then capturing the packets via wireshark, we observe the following:

No.	Time	Source	Destination	Protocol	Length	Info
5	1.457668	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=280/6145, ttl=1 (no response found!)
6	1.459540	192.168.0.1	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
7	1.460796	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=281/6401, ttl=1 (no response found!)
8	1.462116	192.168.0.1	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
9	1.463094	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=282/6657, ttl=1 (no response found!)
10	1.464502	192.168.0.1	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
51	7.409542	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=283/6913, ttl=2 (no response found!)
52	7.411495	192.168.1.251	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
53	7.414912	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=284/7169, ttl=2 (no response found!)
54	7.416775	192.168.1.251	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
55	7.419179	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=285/7425, ttl=2 (no response found!)
56	7.420610	192.168.1.251	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
60	7.844865	192.168.1.251	192.168.0.115	ICMP	120	Destination unreachable (Port unreachable)
70	9.356204	192.168.1.251	192.168.0.115	ICMP	120	Destination unreachable (Port unreachable)
75	10.872650	192.168.1.251	192.168.0.115	ICMP	120	Destination unreachable (Port unreachable)
97	13.387632	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=286/7681, ttl=3 (no response found!)
142	16.955291	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=287/7937, ttl=3 (no response found!)
181	20.958991	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=288/8193, ttl=3 (no response found!)
246	24.963192	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=289/8449, ttl=4 (no response found!)
248	25.015801	59.185.210.193	192.168.0.115	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
249	25.018769	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=290/8705, ttl=4 (no response found!)
250	25.023154	59.185.210.193	192.168.0.115	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
251	25.026220	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=291/8961, ttl=4 (no response found!)
252	25.030947	59.185.210.193	192.168.0.115	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
257	26.052367	192.168.0.115	172.217.167.164	ICMP	106	[Echo (ping) request id=0x0001, seq=292/9217, ttl=5 (no response found!)]
258	26.056698	59.185.210.194	192.168.0.115	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
259	26.059541	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=293/9473, ttl=5 (no response found!)
260	26.063643	59.185.210.194	192.168.0.115	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
261	26.066445	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=294/9729, ttl=5 (no response found!)
262	26.069692	59.185.210.194	192.168.0.115	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
266	27.082607	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=295/9985, ttl=6 (no response found!)
267	27.086939	74.125.51.205	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
268	27.090292	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=296/10241, ttl=6 (no response found!)
269	27.099327	74.125.51.205	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
270	27.102048	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=297/10497, ttl=6 (no response found!)
271	27.114075	74.125.51.205	192.168.0.115	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
357	33.604069	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=298/10753, ttl=7 (no response found!)
358	33.610722	209.85.246.51	192.168.0.115	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
359	33.613122	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=299/11009, ttl=7 (no response found!)
360	33.619215	209.85.246.51	192.168.0.115	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
361	33.622504	192.168.0.115	172.217.167.164	ICMP	106	Echo (ping) request id=0x0001, seq=300/11265, ttl=7 (no response found!)

Here we can observe that my host is sending ICMP echo request packets with increasing TTL and the routers are responding with "TTL Exceeded" packets.

We also see that all packets sent from my IP (192.168.0.115) have destination IP address (the IP address found by DNS lookup of [www.google.com](http://www.google.com)).

### Protocol Info

Since we used the **-I** flag, the protocol used is ICMP.

Layer	Protocol Used
Network	ICMP

### 3. dig

We know that `dig` uses DNS protocol, so we can add "dns" filter in wireshark.

Running `dig www.google.com` in terminal and then capturing the packets via wireshark, we observe the following:

No.	Time	Source	Destination	Protocol	Length	Info
20	2.693392	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x2e19 A www.google.com
21	2.693604	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x0590 AAAA www.google.com
22	2.697202	192.168.0.1	192.168.0.115	DNS	90	Standard query response 0x2e19 A www.google.com A 172.217.167.164
23	2.699146	192.168.0.1	192.168.0.115	DNS	102	Standard query response 0x0590 AAAA www.google.com AAAA 2404:6800:4009:829::2004

DNS request:

No.	Time	Source	Destination	Protocol	Length	Info
20	2.693392	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x2e19 A www.google.com
21	2.693604	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x0590 AAAA www.google.com
22	2.697202	192.168.0.1	192.168.0.115	DNS	90	Standard query response 0x2e19 A www.google.com A 172.217.167.164
23	2.699146	192.168.0.1	192.168.0.115	DNS	102	Standard query response 0x0590 AAAA www.google.com AAAA 2404:6800:4009:829::2004

```

> Frame 20: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{A6D5012E-CD41-4702-B6F5-74CE307F17F8}, id 0
> Ethernet II, Src: IntelCor_a0:4e:11 (a4:c3:f0:a0:4e:11), Dst: TendaTec_09:95:c0 (50:2b:73:09:95:c0)  Query was made to my DNS Server
  Internet Protocol Version 4, Src: 192.168.0.115, Dst: 192.168.0.1  whose IP address in 192.168.0.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSFP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x78ab (30891)
    Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x0000 [validation disabled]
      [Header checksum status: Unverified]
    Source Address: 192.168.0.115
    Destination Address: 192.168.0.1
  > User Datagram Protocol, Src Port: 64306, Dst Port: 53  DNS uses port 53
    Source Port: 64306
    Destination Port: 53
    Length: 40
    Checksum: 0x81fe [unverified]
      [Checksum Status: Unverified]
      [Stream index: 3]
    > [Timestamps]
      UDP payload (32 bytes)
  > Domain Name System (query)
    Transaction ID: 0x2e19
    > Flags: 0x0100 Standard query
      Questions: 1
      Answer RRs: 0
      Authority RRs: 0
      Additional RRs: 0
    > Queries
      > www.google.com: type A, class IN  A Record Query
      [Response In: 22]

```

- Query made from DNS Server 192.168.0.1
- Query: "www.google.com: type A, class IN"

## DNS response:

dns

No.	Time	Source	Destination	Protocol	Length	Info
20	2.693392	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x2e19 A www.google.com
21	2.693604	192.168.0.115	192.168.0.1	DNS	74	Standard query 0x0590 AAAA www.google.com
22	2.697202	192.168.0.1	192.168.0.115	DNS	98	Standard query response 0x2e19 A www.google.com A 172.217.167.164
23	2.699146	192.168.0.1	192.168.0.115	DNS	102	Standard query response 0x0590 AAAA www.google.com AAAA 2404:6800:4009:829::2004

```

> Frame 22: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface \Device\NPF_{A6D5012E-CD41-4702-B6F5-74CE307F17F8}, id 0
> Ethernet II, Src: TendaTec_09:95:c0 (50:2b:73:09:95:c0), Dst: IntelCor_a0:4e:11 (a4:c3:f0:a0:4e:11)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.115
> User Datagram Protocol, Src Port: 53, Dst Port: 64306
    Source Port: 53
    Destination Port: 64306
    Length: 56
    Checksum: 0x2f9c [unverified]
        [Checksum Status: Unverified]
        [Stream index: 3]
    > [Timestamps]
        UDP payload (48 bytes)
> Domain Name System (response)
    Transaction ID: 0x2e19
    > Flags: 0x8100 Standard query response, No error
        Questions: 1
        Answer RRs: 1
        Authority RRs: 0
        Additional RRs: 0
    > Queries
        > www.google.com: type A, class IN
    > Answers
        > www.google.com: type A, class IN, addr 172.217.167.164
            Name: www.google.com
            Type: A (Host Address) (1)
            Class: IN (0x0001)
            Time to live: 226 (3 minutes, 46 seconds)
            Data length: 4
            Address: 172.217.167.164
            [Request In: 20]
            [Time: 0.003810000 seconds]

```

Here we can see the IP Address of [www.google.com](http://www.google.com)  
which is 172.217.167.164

- We received an A record for [www.google.com](http://www.google.com) with address 172.217.167.164

## Protocol Info

- Protocol used here is DNS

Layer	Protocol Used
Application	DNS
Transport	UDP
Network	IP

## 4. arp

arp maintains a table of mac addresses and corresponding IP addresses. So when we run `arp` command, it sends a broadcast packet to all the hosts in the network.

```

No. Time Source Destination Protocol Length Info
1 0.000000 EdimaxTe_a8:ac:21 Broadcast ARP 42 Who has 192.168.0.1? Tell 192.168.0.130

> Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface \Device\NPF_{A6D5012E-CD41-4702-B6F5-74CE307F17F8}, id 0
  ✓ Ethernet II, Src: EdimaxTe_a8:ac:21 (74:da:38:a8:ac:21), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
    > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    > Source: EdimaxTe_a8:ac:21 (74:da:38:a8:ac:21) Source is My Machine
    Type: ARP (0x0806) ARP (0x0806)
  ✓ Address Resolution Protocol (request) ARP Protocol
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
      Sender MAC address: EdimaxTe_a8:ac:21 (74:da:38:a8:ac:21)
      Sender IP address: 192.168.0.130
      Target MAC address: Broadcast (ff:ff:ff:ff:ff:ff)
      Target IP address: 192.168.0.1
  
```

- Destination: Broadcast (`ff:ff:ff:ff:ff:ff`)

### Protocol Info

- Protocol: ARP

Layer	Protocol Used
Network	ARP

## 5. wget

wget makes a GET HTTP(S) request to a server and downloads the file. Running `wget www.google.com` in terminal and then capturing the packets via wireshark, we observe the following:

No.	Time	Source	Destination	Protocol	Length	Info
74	1.601919	192.168.0.115	142.250.66.3	TCP	66	51079 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
75	1.606835	142.250.66.3	192.168.0.115	TCP	66	443 → 51079 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM=1 WS=256
76	1.606913	192.168.0.115	142.250.66.3	TCP	54	51079 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
77	1.607454	192.168.0.115	142.250.66.3	TLSv1.3	325	Client Hello
78	1.617460	142.250.66.3	192.168.0.115	TCP	54	443 → 51079 [ACK] Seq=1 Ack=272 Win=66816 Len=0
79	1.680256	142.250.66.3	192.168.0.115	TLSv1.3	1466	Server Hello, Change Cipher Spec
80	1.680256	142.250.66.3	192.168.0.115	TCP	1466	443 → 51079 [PSH, ACK] Seq=1413 Ack=272 Win=66816 Len=1412 [TCP segment of a reassembled PDU]
81	1.680256	142.250.66.3	192.168.0.115	TCP	1466	443 → 51079 [ACK] Seq=2825 Ack=272 Win=66816 Len=1412 [TCP segment of a reassembled PDU]
82	1.688256	142.250.66.3	192.168.0.115	TLSv1.3	165	Application Data
83	1.680396	192.168.0.115	142.250.66.3	TCP	54	51079 → 443 [ACK] Seq=272 Ack=4348 Win=262144 Len=0
84	1.683047	192.168.0.115	142.250.66.3	TLSv1.3	134	Change Cipher Spec, Application Data
85	1.683320	192.168.0.115	142.250.66.3	TLSv1.3	428	Application Data
86	1.687290	142.250.66.3	192.168.0.115	TCP	54	443 → 51079 [ACK] Seq=4348 Ack=352 Win=66816 Len=0
87	1.687290	142.250.66.3	192.168.0.115	TCP	54	443 → 51079 [ACK] Seq=4348 Ack=726 Win=67840 Len=0
88	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
89	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	620	Application Data
90	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
91	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
92	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
93	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
94	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
95	1.690941	142.250.66.3	192.168.0.115	TLSv1.3	1466	Application Data
96	1.691022	192.168.0.115	142.250.66.3	TCP	54	51079 → 443 [ACK] Seq=726 Ack=14798 Win=262144 Len=0
97	1.691091	142.250.66.3	192.168.0.115	TLSv1.3	900	Application Data
98	1.691111	192.168.0.115	142.250.66.3	TCP	54	51079 → 443 [ACK] Seq=726 Ack=15644 Win=261120 Len=0

- First Packet was sent with **SYN** flag set
- Second Packet was sent with **SYN-ACK** flag set
- Finally, Third Packet was sent with **ACK** flag set
- Now the handshaking is done, data transfer starts

Looking at a packet of Data Transfer we see:

```
...N·P+s ... E ... z ...)·B ... s ... k·4·4IP... ... S·y.=Xf...
·1...ae·$n· VHnd}·1...LH·Zh||i·c...<·8@... s ... @!...|.*...}b...@}
1H...;...x...@+·M `t%|0*·4..._S...3...:p·V...,c k·4...[...
·^ZY·5G...>c..._\...2k ^G...~g...E...FCP·C...F...~Z v...`8^·i·4·3KY·Z
...N...^ZF...Y*a...G...m...*... f...i·m8C...2...).m...8...*...%·wD...0·0...$g...
·T·h...[...D·y...a...ho...3@#·5·o...q...Y...rI...nY...n·P...=·d·z.../
...A...@...n&...ny...d·8·J*...S...IU...r..."\...z...Y...Ds...I·0...J,...06...
· *6...o...*6...xjqE)...;...G%...f...@l...U...t...j...*...A...i...>..._v){...G+...f...Fc...7...a...|...
5...&...]*B...x...0...vG...d...w$... /<...f...5...8.../4ot...}xw...!
K[...ck...3>...(*...I"...c...8...].UM...dC...?...n...B...#...t...j.../...d...)|...Y...3...
·v...x...j9...L)R...q[w...$...X.../x...
...*...vca...;...Q...C...a...en9...w...\\%...gD...:...8...?...h...G...
...Q...r...i...g...z...lD...q...1...{...u...Mf...q...|...n...h...o...B&
H...g...,.\\f...M...<+...v][...z>80...K1...9...=g...Z...7af
"....J...).H...9...)]...9...%)...>...D1...f...
Yq8...$?...+...K...v\6...@...8...y...);...9...1...(...YD...7...z4...1...}{...<k...o.../
`...6...Yq...p...Zt...D...0Y008...g...&...v...o...K...?...8...Y...a...J...C...J1...T...
·B..._...x...6...0vb...*{...j...B...Xt...].%...Y...i...>...>>f...
;...2x...smKI...+...6...([...H...T...<...p...NG?z|
q...y...t...g...u...So...y...W...i}...K...G=...&...g...gY...v|
...3...>...+lBO)5H...h...f...5..._...F...o...A...U...q...z...)(t...c...m...WC...4
```

- here all data is gibberish, because it was end-to-end encrypted. The request was made through HTTPS protocol which makes it impossible for sniffers like wireshark to decode the data.

## Protocol Info

- Application Layer Protocol: HTTPS
- Transport Layer Protocol: TCP
- Network Layer Protocol: IP

<b>Layer</b>	<b>Protocol Used</b>
Application	HTTPS
Transport	TCP
Network	IP

## Question 2

---

Capture the packets while sending/receiving **telnet** request/response between your computer and a custom server running the **telnet** daemon. What is your observation while analyzing the application layer data?

### Packet Capture

I have set up a telnet daemon in a VM which is in Bridged Mode. The VM's IP is **192.168.0.174**. I have installed telnetd which is actively listening on port 23.

Connecting to VM via **telnet 192.168.0.174** and then capturing the packets via wireshark, we observe the following:

No.	Time	Source	Destination	Protocol	Length	Info
457	15.811016	192.168.0.115	192.168.0.174	TCP	66	55280 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
458	15.811023	192.168.0.115	192.168.0.174	TCP	66	[TCP Out-of-Order] [TCP Port numbers reused] 55280 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
459	15.811488	192.168.0.174	192.168.0.115	TCP	66	23 + 55280 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
460	15.811495	192.168.0.174	192.168.0.115	TCP	66	[TCP Out-of-Order] 23 + 55280 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
461	15.811632	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=1 Ack=1 Win=131328 Len=0
462	15.811640	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 461#1] 55280 → 23 [ACK] Seq=1 Ack=1 Win=131328 Len=0
540	16.469042	192.168.0.174	192.168.0.115	TELNET	66	Telnet Data ...
541	16.469046	192.168.0.174	192.168.0.115	TCP	66	[TCP Retransmission] 23 + 55280 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=12
542	16.469336	192.168.0.115	192.168.0.174	TELNET	60	Telnet Data ...
543	16.469342	192.168.0.115	192.168.0.174	TCP	60	[TCP Retransmission] 55280 → 23 [PSH, ACK] Seq=1 Ack=13 Win=131328 Len=6
544	16.469694	192.168.0.174	192.168.0.115	TCP	60	23 + 55280 [ACK] Seq=13 Ack=7 Win=64256 Len=0
545	16.469698	192.168.0.174	192.168.0.115	TCP	60	[TCP Dup ACK 544#1] 23 + 55280 [ACK] Seq=13 Ack=7 Win=64256 Len=0
546	16.469732	192.168.0.115	192.168.0.174	TELNET	63	Telnet Data ...
547	16.469739	192.168.0.115	192.168.0.174	TCP	63	[TCP Retransmission] 55280 → 23 [PSH, ACK] Seq=7 Ack=13 Win=131328 Len=9
548	16.469989	192.168.0.174	192.168.0.115	TCP	60	23 + 55280 [ACK] Seq=13 Ack=16 Win=64256 Len=0
549	16.469992	192.168.0.174	192.168.0.115	TCP	60	[TCP Dup ACK 548#1] 23 + 55280 [ACK] Seq=13 Ack=16 Win=64256 Len=0
550	16.470237	192.168.0.174	192.168.0.115	TELNET	69	Telnet Data ...
551	16.470241	192.168.0.174	192.168.0.115	TCP	69	[TCP Retransmission] 23 + 55280 [PSH, ACK] Seq=13 Ack=16 Win=64256 Len=15
554	16.470414	192.168.0.115	192.168.0.174	TELNET	63	Telnet Data ...
555	16.470424	192.168.0.115	192.168.0.174	TCP	63	[TCP Retransmission] 55280 → 23 [PSH, ACK] Seq=16 Ack=28 Win=131328 Len=9
558	16.470719	192.168.0.174	192.168.0.115	TCP	60	23 + 55280 [ACK] Seq=28 Ack=25 Win=64256 Len=0
559	16.470722	192.168.0.174	192.168.0.115	TCP	60	[TCP Dup ACK 558#1] 23 + 55280 [ACK] Seq=28 Ack=25 Win=64256 Len=0
560	16.470759	192.168.0.115	192.168.0.174	TELNET	70	Telnet Data ...
563	16.470763	192.168.0.115	192.168.0.174	TCP	70	[TCP Retransmission] 55280 → 23 [PSH, ACK] Seq=25 Ack=28 Win=131328 Len=16
564	16.471044	192.168.0.174	192.168.0.115	TCP	60	23 + 55280 [ACK] Seq=28 Ack=41 Win=64256 Len=0
565	16.471047	192.168.0.174	192.168.0.115	TCP	60	[TCP Dup ACK 564#1] 23 + 55280 [ACK] Seq=28 Ack=41 Win=64256 Len=0

We observe that same three-way TCP handshaking is done between the telnet client and the telnet daemon.

After that every data is passed through the **TELNET** Protocol.

## Observing the TELNET packets

tcp.port == 23						
No.	Time	Source	Destination	Protocol	Length	Info
961	24.025748	192.168.0.115	192.168.0.174	TCP	56	[TCP Retransmission] 55280 → 23 [PSH, ACK] Seq=75 Ack=101 Win=131072 Len=2
962	24.026225	192.168.0.174	192.168.0.115	TCP	60	23 → 55280 [ACK] Seq=101 Ack=77 Win=64256 Len=0
963	24.026233	192.168.0.174	192.168.0.115	TCP	60	[TCP Dup ACK 962#1] 23 → 55280 [ACK] Seq=101 Ack=77 Win=64256 Len=0
964	24.029505	192.168.0.174	192.168.0.115	TELNET	60	Telnet Data ...
965	24.029509	192.168.0.174	192.168.0.115	TCP	60	[TCP Retransmission] 23 → 55280 [PSH, ACK] Seq=101 Ack=77 Win=64256 Len=2
966	24.073038	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=77 Ack=103 Win=131072 Len=0
967	24.073048	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 966#1] 55280 → 23 [ACK] Seq=77 Ack=103 Win=131072 Len=0
975	24.650859	192.168.0.174	192.168.0.115	TELNET	786	Telnet Data ...
976	24.650865	192.168.0.174	192.168.0.115	TCP	786	[TCP Retransmission] 23 → 55280 [PSH, ACK] Seq=103 Ack=77 Win=64256 Len=732
977	24.690502	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=77 Ack=835 Win=130560 Len=0
978	24.690506	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 977#1] 55280 → 23 [ACK] Seq=77 Ack=835 Win=130560 Len=0
979	24.690830	192.168.0.174	192.168.0.115	TELNET	104	Telnet Data ...
980	24.690834	192.168.0.174	192.168.0.115	TCP	104	[TCP Retransmission] 23 → 55280 [PSH, ACK] Seq=835 Ack=77 Win=64256 Len=50
986	24.734209	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=77 Ack=885 Win=130304 Len=0
987	24.734215	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 986#1] 55280 → 23 [ACK] Seq=77 Ack=885 Win=130304 Len=0
988	24.738899	192.168.0.174	192.168.0.115	TELNET	150	Telnet Data ...
989	24.738904	192.168.0.174	192.168.0.115	TCP	100	[TCP Retransmission] 23 → 55280 [PSH, ACK] Seq=885 Ack=77 Win=64256 Len=96
990	24.781234	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=77 Ack=981 Win=130304 Len=0
991	24.781239	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 990#1] 55280 → 23 [ACK] Seq=77 Ack=981 Win=130304 Len=0
1401	30.275136	192.168.0.115	192.168.0.174	TCP	54	95280 → 23 [FIN, ACK] Seq=77 Ack=981 Win=1049600 Len=0
1402	30.275155	192.168.0.115	192.168.0.174	TCP	54	[TCP Out-Of-Order] 55280 → 23 [FIN, ACK] Seq=77 Ack=981 Win=1049600 Len=0
1403	30.276071	192.168.0.174	192.168.0.115	TCP	60	23 → 55280 [FIN, ACK] Seq=981 Ack=78 Win=64256 Len=0
1404	30.276076	192.168.0.174	192.168.0.115	TCP	60	[TCP Out-Of-Order] 23 → 55280 [FIN, ACK] Seq=981 Ack=78 Win=64256 Len=0
1405	30.276135	192.168.0.115	192.168.0.174	TCP	54	55280 → 23 [ACK] Seq=78 Ack=982 Win=1049600 Len=0
1406	30.276139	192.168.0.115	192.168.0.174	TCP	54	[TCP Dup ACK 1405#1] 55280 → 23 [ACK] Seq=78 Ack=982 Win=1049600 Len=0

[Next Sequence Number: 885 (relative sequence number)]
Acknowledgment Number: 77 (relative ack number)
Acknowledgment number (raw): 2644118871
0101 .... = Header Length: 20 bytes (5)
> Flags: 0x018 (PSH, ACK)
Window: 502
[Calculated window size: 64256]
[Window size scaling factor: 128]
Checksum: 0xe027 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]
TCP payload (56 bytes)
▼ Telnet
Data: Last login: Wed Feb 9 11:17:56 UTC 2022 on tty1\r\nn

0000 a4 c3 f0 a0 4e 11 a4 c3 f0 a0 4e 11 08 00 45 10 .....N.....N...E.
0010 00 5a 49 12 40 00 48 06 6f 0a c0 a8 00 ae c0 a8 .ZI @@ o .....
0020 00 73 00 17 d7 f0 e8 dc ee a9 9d 9a 0d 57 50 18 .S..... .WP.
0030 01 f6 e0 27 00 00 4c 61 73 74 20 6c 6f 67 69 6e ....La st login:
0040 3a 20 57 65 64 20 46 65 62 20 20 39 20 31 31 3a : Wed Fe b 9 11:
0050 31 37 3a 35 36 20 55 54 43 20 32 30 32 32 20 6f 17:56 UT C 2022 o
0060 6e 20 74 74 79 31 0d 0a n tty1..

TELNET Protocol used to send the data

Data is not encrypted

- NOTE: Data Transfer is not encrypted and can be easily seen by sniffing softwares like wireshark.

## Question 3

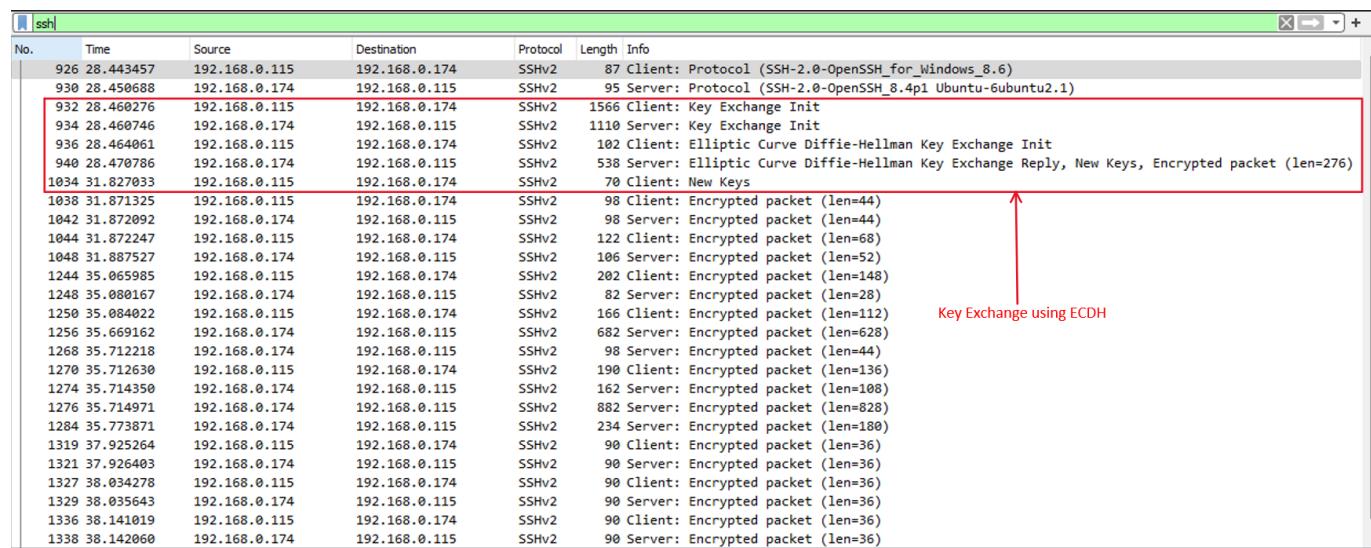
---

Capture the packets while sending/receiving **ssh** request/response between your computer and one of the department servers. What is your observation while analyzing the application layer data?

### Packet Capture

**SSH** uses asymmetric cryptography to establish a shared secret key then uses the symmetric cryptography with the shared key for the remote session.

So, the first step of ssh will be to establish a shared secret key. Here we can see the server and the client initiating the key exchange via Elliptic Curve Diffie-Hellman (ECDH) key exchange algorithm.



Key Exchange using ECDH

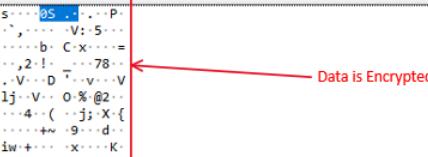
No.	Time	Source	Destination	Protocol	Length	Info
926	28.443457	192.168.0.115	192.168.0.174	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_8.6)
930	28.450688	192.168.0.174	192.168.0.115	SSHv2	95	Server: Protocol (SSH-2.0-OpenSSH_8.4p1 Ubuntu-6ubuntu2.1)
932	28.460276	192.168.0.115	192.168.0.174	SSHv2	1566	Client: Key Exchange Init
934	28.460746	192.168.0.174	192.168.0.115	SSHv2	1110	Server: Key Exchange Init
936	28.464061	192.168.0.115	192.168.0.174	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
940	28.470786	192.168.0.174	192.168.0.115	SSHv2	538	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet (len=276)
1034	31.827033	192.168.0.115	192.168.0.174	SSHv2	70	Client: New Keys
1038	31.871325	192.168.0.115	192.168.0.174	SSHv2	98	Client: Encrypted packet (len=44)
1042	31.872092	192.168.0.174	192.168.0.115	SSHv2	98	Server: Encrypted packet (len=44)
1044	31.872247	192.168.0.115	192.168.0.174	SSHv2	122	Client: Encrypted packet (len=68)
1048	31.887527	192.168.0.174	192.168.0.115	SSHv2	106	Server: Encrypted packet (len=52)
1244	35.065985	192.168.0.115	192.168.0.174	SSHv2	202	Client: Encrypted packet (len=148)
1248	35.080167	192.168.0.174	192.168.0.115	SSHv2	82	Server: Encrypted packet (len=28)
1250	35.084022	192.168.0.115	192.168.0.174	SSHv2	166	Client: Encrypted packet (len=112)
1256	35.669162	192.168.0.174	192.168.0.115	SSHv2	682	Server: Encrypted packet (len=628)
1268	35.712218	192.168.0.174	192.168.0.115	SSHv2	98	Server: Encrypted packet (len=44)
1270	35.712630	192.168.0.115	192.168.0.174	SSHv2	190	Client: Encrypted packet (len=136)
1274	35.714350	192.168.0.174	192.168.0.115	SSHv2	162	Server: Encrypted packet (len=108)
1276	35.714971	192.168.0.174	192.168.0.115	SSHv2	882	Server: Encrypted packet (len=828)
1284	35.773871	192.168.0.174	192.168.0.115	SSHv2	234	Server: Encrypted packet (len=180)
1319	37.925264	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1321	37.926403	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1327	38.034278	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1329	38.035643	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1336	38.141019	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1338	38.142060	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)

After this the symmetric key is agreed upon the client and the server and all the remote session data is encrypted using the shared key.

No.	Time	Source	Destination	Protocol	Length	Info
936	28.464061	192.168.0.115	192.168.0.174	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
940	28.470786	192.168.0.174	192.168.0.115	SSHv2	538	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet (len=276)
1034	31.827033	192.168.0.115	192.168.0.174	SSHv2	70	Client: New Keys
1038	31.871325	192.168.0.115	192.168.0.174	SSHv2	98	Client: Encrypted packet (len=44)
1042	31.872092	192.168.0.174	192.168.0.115	SSHv2	98	Server: Encrypted packet (len=44)
1044	31.872247	192.168.0.115	192.168.0.174	SSHv2	122	Client: Encrypted packet (len=68)
1048	31.887527	192.168.0.174	192.168.0.115	SSHv2	106	Server: Encrypted packet (len=52)
1244	35.065985	192.168.0.115	192.168.0.174	SSHv2	202	Client: Encrypted packet (len=148)
1248	35.080167	192.168.0.174	192.168.0.115	SSHv2	82	Server: Encrypted packet (len=28)
1250	35.084022	192.168.0.115	192.168.0.174	SSHv2	166	Client: Encrypted packet (len=112)
1256	35.669162	192.168.0.174	192.168.0.115	SSHv2	682	Server: Encrypted packet (len=628)
1268	35.712218	192.168.0.174	192.168.0.115	SSHv2	98	Server: Encrypted packet (len=44)
1270	35.712630	192.168.0.115	192.168.0.174	SSHv2	190	Client: Encrypted packet (len=136)
1274	35.714350	192.168.0.174	192.168.0.115	SSHv2	162	Server: Encrypted packet (len=108)
1276	35.714971	192.168.0.174	192.168.0.115	SSHv2	882	Server: Encrypted packet (len=828)
1284	35.773871	192.168.0.174	192.168.0.115	SSHv2	234	Server: Encrypted packet (len=188)
1319	37.925264	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1321	37.926403	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1327	38.034278	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1329	38.035643	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1336	38.141019	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1338	38.142060	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1343	38.257870	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1345	38.259231	192.168.0.174	192.168.0.115	SSHv2	90	Server: Encrypted packet (len=36)
1354	38.722673	192.168.0.115	192.168.0.174	SSHv2	90	Client: Encrypted packet (len=36)
1356	38.723827	192.168.0.174	192.168.0.115	SSHv2	106	Server: Encrypted packet (len=52)

Sequence Number (raw): 810757649  
[Next Sequence Number: 3494 (relative sequence number)]  
Acknowledgment Number: 2118 (relative ack number)  
Acknowledgment number (raw): 3123617211  
0101 .... = Header Length: 20 bytes (5)  
> Flags: 0x018 (PSH, ACK)  
Window: 501  
[Calculated window size: 64128]  
[Window size scaling factor: 128]  
Checksum: 0x602c [unverified]  
[Checksum Status: Unverified]  
Urgent Pointer: 0  
> [Timestamps]  
> [SEQ/ACK analysis]  
TCP payload (180 bytes)  
> SSH Protocol

0020	00 73 00 16 da 9a <b>30 53 2e 11</b> ba 2e 9d bb 50 18	s---05 .. .P `,...,..V:5... ....b C x ...=
0030	01 f5 60 2c 00 00 ee b4 ae 56 3a 11 35 cf 12 ea	,2!...78... .V...D T...v...V
0040	1d c0 09 1d 97 ec 62 e2 43 a8 78 11 f4 ae 83 3d	l1..V.. 0 %@2... ..4..( ..j; X {
0050	9d e4 c8 2c 32 fe 21 97 5f 17 b2 af 37 38 eb 1c	....w...9...d... iw+....x...K-
0060	89 2e 02 56 1d 9a 99 44 27 af 9b 76 c7 c3 b2 56	
0070	07 6c 6a a0 8a 56 1b 19 4f 09 25 84 40 32 1a 9d	
0080	1f e3 96 a4 34 d3 a4 28 9d b5 6a 3b f5 58 d4 7b	
0090	8c fe d3 1b 9f e4 2b 7e 15 39 fb 99 d6 64 0f c9	
00a0	b9 69 77 1f 2b b8 f4 ce 8d 78 a0 c5 94 b5 4b e2	



## Question 4

Enter the URL: <http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html> and capture packets using Wireshark. After your browser has displayed the INTRO-wireshark-file1.html page (it is a simple one line of congratulations), stop Wireshark's packet capture. Answer the following from the packets captured:

- How long did it take from when the HTTP GET message was sent until the HTTP OK reply was received?
- What is the Internet address of the **gaia.cs.umass.edu**? What is the Internet address of your computer? Support your answer with an appropriate screenshot from your computer.

### 1. How long did it take from when the HTTP GET message was sent until the HTTP OK reply was received?

The first GET request was made in 0.504310921 s

[Time since first frame in this TCP stream: 0.504310921 seconds]

Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.0000000000	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1460 [TCP segment of a reassembled PDU]	
2	0.0000000058	192.168.0.115	20.44.229.112	TLSv1.2	234 Application Data	
3	0.545320095	192.168.0.136	192.168.0.1	DNS	77 Standard query 0x21f0 A gaia.cs.umass.edu	
4	0.545548971	192.168.0.136	192.168.0.1	DNS	77 Standard query 0x8eed AAAA gaia.cs.umass.edu	
5	0.565352971	192.168.0.1	192.168.0.136	DNS	93 Standard query response 0x21f0 A gaia.cs.umass.edu A 128.119.245.12	
6	0.565353023	192.168.0.1	192.168.0.136	DNS	77 Standard query response 0x8eed AAAA gaia.cs.umass.edu	
7	0.565839839	192.168.0.136	128.119.245.12	TCP	74 40126 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=479973315 TSecr=0 WS=128	
8	0.661472285	Espresso_ac:f9:21	Broadcast	ARP	60 ARP Announcement for 192.168.0.156	
9	0.971869305	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=1641 Ack=391 Win=511 Len=1460 [TCP segment of a reassembled PDU]	
10	0.971869348	192.168.0.115	20.44.229.112	TLSv1.2	234 Application Data	
11	1.069838891	128.119.245.12	192.168.0.136	TCP	74 80 → 40126 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1412 SACK_PERM=1 TStamp=4103218773 TSecr=479973315 WS=128	
12	1.069869961	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TStamp=479973819 TSecr=4103218773	
13	1.070150760	192.168.0.136	128.119.245.12	HTTP	188 GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1	
14	1.276183975	TendaTec_09:95:c0	Broadcast	ARP	60 Who has 192.168.0.134? Tell 192.168.0.1	
15	1.397507978	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819	
16	1.397508028	128.119.245.12	192.168.0.136	TCP	66 [TCP Dup ACK 15#1] 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819	
17	1.397508050	128.119.245.12	192.168.0.136	HTTP	448 HTTP/1.1 200 OK (text/html)	
18	1.397532056	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974146 TSecr=4103219186	
19	1.397893267	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [FIN, ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974147 TSecr=4103219186	
20	1.787087390	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [FIN, ACK] Seq=383 Ack=124 Win=29056 Len=0 TStamp=4103219512 TSecr=479974147	
21	1.787112455	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=124 Ack=384 Win=64128 Len=0 TStamp=479974536 TSecr=4103219512	
22	1.792599104	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=3281 Ack=781 Win=509 Len=1460 [TCP segment of a reassembled PDU]	
23	1.792599149	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data	
24	1.995942007	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=4920 Ack=1171 Win=508 Len=1460 [TCP segment of a reassembled PDU]	
25	1.995942067	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data	
26	2.199771226	192.168.0.156	255.255.255.255	UDP	230 49154 → 6667 Len=188	
27	2.331082781	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=6559 Ack=1561 Win=512 Len=1460 [TCP segment of a reassembled PDU]	
28	2.331082829	192.168.0.115	20.44.229.112	TLSv1.2	237 Application Data	
[Window size scaling factor: 128] Checksum: 0x3755 [unverified] [Checksum Status: Unverified] Urgent Pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps [Timestamps] [Time since first frame in this TCP stream: 0.504310921 seconds] [Time since previous frame in this TCP stream: 0.000000022 seconds]						

The first **GET** request was made in 0.831668211 s

[Time since first frame in this TCP stream: 0.83166821 seconds]

15 1.397507978	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819	HTTP Response Received	
16 1.397508028	128.119.245.12	192.168.0.136	TCP	66 [TCP Dup ACK 15#1] 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819		
17 1.397508050	128.119.245.12	192.168.0.136	HTTP	448 HTTP/1.1 200 OK (text/html)		
18 1.397532056	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974146 TSecr=4103219186		
19 1.397893267	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [FIN, ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974147 TSecr=4103219186		
20 1.787087390	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [FIN, ACK] Seq=383 Ack=124 Win=29056 Len=0 TStamp=4103219512 TSecr=479974147		
21 1.787112455	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=124 Ack=384 Win=64128 Len=0 TStamp=479974536 TSecr=4103219512		
22 1.792599104	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=3281 Ack=781 Win=509 Len=1460 [TCP segment of a reassembled PDU]		
23 1.792599149	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data		
24 1.995942007	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=4920 Ack=1171 Win=508 Len=1460 [TCP segment of a reassembled PDU]		
25 1.995942067	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data		
26 2.199771226	192.168.0.156	255.255.255.255	UDP	230 49154 → 6667 Len=188		
27 2.331082781	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=6559 Ack=1561 Win=512 Len=1460 [TCP segment of a reassembled PDU]		
28 2.331082829	192.168.0.115	20.44.229.112	TLSv1.2	237 Application Data		
[Window size scaling factor: 128] Checksum: 0x9336 [unverified] [Checksum Status: Unverified] Urgent Pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps [Timestamps] [Time since first frame in this TCP stream: 0.831668211 seconds] [Time since previous frame in this TCP stream: 0.000000022 seconds]						

So Time Taken = 0.831668211 - 0.504310921 = 0.32735729 s

2. What is the Internet address of the **gaia.cs.umass.edu**? What is the Internet address of your computer? Support your answer with an appropriate screenshot from your computer.

12 1.0000000001	192.168.0.136	128.119.245.12	TCP	66 40120 → 66 [ACK] Seq=1 ACK=1 Win=0 TStamp=479973819	Source and Destination Address	
13 1.070150760	192.168.0.136	128.119.245.12	HTTP	188 GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1		
14 1.276183975	TendaTec_09:95:c0	Broadcast	ARP	60 Who has 192.168.0.134? Tell 192.168.0.1		
15 1.397507978	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819		
16 1.397508028	128.119.245.12	192.168.0.136	TCP	66 [TCP Dup ACK 15#1] 80 → 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TStamp=4103219185 TSecr=479973819		
17 1.397508050	128.119.245.12	192.168.0.136	HTTP	448 HTTP/1.1 200 OK (text/html)		
18 1.397532056	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974146 TSecr=4103219186		
19 1.397893267	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [FIN, ACK] Seq=123 Ack=383 Win=64128 Len=0 TStamp=479974147 TSecr=4103219186		
20 1.787087390	128.119.245.12	192.168.0.136	TCP	66 80 → 40126 [FIN, ACK] Seq=383 Ack=124 Win=29056 Len=0 TStamp=4103219512 TSecr=479974147		
21 1.787112455	192.168.0.136	128.119.245.12	TCP	66 40126 → 80 [ACK] Seq=124 Ack=384 Win=64128 Len=0 TStamp=479974536 TSecr=4103219512		
22 1.792599104	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=3281 Ack=781 Win=509 Len=1460 [TCP segment of a reassembled PDU]		
23 1.792599149	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data		
24 1.995942007	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=4920 Ack=1171 Win=508 Len=1460 [TCP segment of a reassembled PDU]		
25 1.995942067	192.168.0.115	20.44.229.112	TLSv1.2	233 Application Data		
26 2.199771226	192.168.0.156	255.255.255.255	UDP	230 49154 → 6667 Len=188		
27 2.331082781	192.168.0.115	20.44.229.112	TCP	1514 49360 → 443 [ACK] Seq=6559 Ack=1561 Win=512 Len=1460 [TCP segment of a reassembled PDU]		
28 2.331082829	192.168.0.115	20.44.229.112	TLSv1.2	237 Application Data		
[Window size scaling factor: 128] Checksum: 0x0000 [unverified] [Checksum Status: Unverified] Urgent Pointer: 0 Options: (12 bytes), No-fragment Flags: 0x0000 0000 0000 = Fragment Offset: 0 Time to Live: 64 Protocol: TCP (6) Header Checksum: 0x53cd [validation disabled] [Header checksum status: Unverified] Source Address: 192.168.0.136 Destination Address: 128.119.245.12						

- IP address of **gaia.cs.umass.edu** is Destination Address: 128.119.245.12
- IP address of my computer is Source Address: 192.168.0.136

## Question 5

---

Start the **Wireshark** packet capturing service. Enter the URL: <https://www.gmail.com/> on your browser and sign-in to your gmail account by providing credentials (Username/Password). Answer the following from the captured packets:

1. Is there any difference in the application layer protocol?
2. How it is different from the **HTTP** data you analysed in the above problem?

### 1. Is there any difference in the application layer protocol?

Yes, the current application layer protocol is **TLS**. A primary use case of **TLS** is encrypting the communication between web applications and servers, such as web browsers loading a website.

But, in the previous case it was just **HTTP**.

### 2. How it is different from the **HTTP** data you analysed in the above problem?

Previously, the protocol used was simple HTTP. As a result sniffers like wireshark could decode the data. Here is an example:

```

12 1.000000000 192.168.0.136 128.119.245.12 TCP 66 40126 66 [ACK] Seq=1 ACK=1 Win=64256 Len=0 TSval=4103219185 TSecr=4103219773
13 1.070156768 192.168.0.136 128.119.245.12 HTTP 188 GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1
14 1.276183975 TendaTec_09:95:c0 Broadcast ARP 60 Who has 192.168.0.134? Tell 192.168.0.1
15 1.397507978 128.119.245.12 192.168.0.136 TCP 66 80 - 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TSval=4103219185 TSecr=479973819
16 1.397508028 128.119.245.12 192.168.0.136 TCP 66 [TCP Dup ACK 15#1] 80 - 40126 [ACK] Seq=1 Ack=123 Win=29056 Len=0 TSval=4103219185 TSecr=479973819
17 1.397508050 128.119.245.12 192.168.0.136 HTTP 448 HTTP/1.1 200 OK (text/html)
18 1.397532056 192.168.0.136 128.119.245.12 TCP 66 40126 - 80 [ACK] Seq=123 Ack=383 Win=64128 Len=0 TSval=479974146 TSecr=4103219186
19 1.397893267 192.168.0.136 128.119.245.12 TCP 66 40126 - 80 [FIN, ACK] Seq=123 Ack=383 Win=64128 Len=0 TSval=479974147 TSecr=4103219186
20 1.787687390 128.119.245.12 192.168.0.136 TCP 66 80 - 40126 [FIN, ACK] Seq=383 Ack=124 Win=29056 Len=0 TSval=4103219512 TSecr=479974147
21 1.787112455 192.168.0.136 128.119.245.12 TCP 66 48126 - 80 [ACK] Seq=124 Ack=384 Win=64128 Len=0 TSval=479974536 TSecr=4103219512
    
```

> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)  
Total Length: 174  
Identification: 0xaafc8 (45000)  
Flags: 0x40, Don't fragment  
...0 0000 0000 0000 = Fragment Offset: 0  
Time to Live: 64  
Protocol: TCP (6)  
Header Checksum: 0x53cd [validation disabled]  
[Header checksum status: Unverified]  
Source Address: 192.168.0.136  
Destination Address: 128.119.245.12  
Transmission Control Protocol, Src Port: 40126, Dst Port: 80, Seq: 1, Ack: 1, Len: 122

Hypertext Transfer Protocol  
- GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1\r\n  
> [Expert Info (Chat/Sequence): GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1\r\n]  
Request Method: GET  
Request URI: /wireshark-labs/INTRO-wireshark-file1.html  
Request Version: HTTP/1.1

```

1940 26 55 47 45 54 20 2f 77 69 72 65 73 68 61 72 6b
1950 2d 6c 01 62 73 2f 49 4e 54 52 4f 2d 77 69 72 65
1960 73 68 61 72 6b 2d 66 69 6c 65 31 2e 68 74 6d 6c
1970 20 48 54 55 58 2f 31 2e 31 0d 0a 48 6f 73 74 3a
1980 20 67 61 69 61 2e 63 73 2e 75 6d 61 73 73 2e 65
1990 64 75 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20
19a0 63 75 72 6c 2f 37 2e 37 39 2e 31 0d 0a 41 63 63
19b0 65 70 74 3a 20 2a 2f 2a 0d 0a 0d 0a
    
```

&GET /w ireshark  
-labs/IN TRO-wire  
shark-fi le1.html

HTTP/1.1..Host:  
gaiac.cs .umass.e  
du..User-Agent:  
curl/7.7 9.1..Acc  
ept: \*/\* ....

Plain Text

But <https://www.gmail.com/> is using **HTTPS** with **TLS**. So the data will be encrypted.

```

20 0.420134287 192.168.0.130 172.217.100.37 TCP 66 60810 -> 443 [ACK] Seq=795 Ack=4967 Win=64128 Len=0 TSval=2414623297 TSecr=1251533554
21 0.426635483 192.168.0.136 172.217.100.37 TLSv1.3 97 Application Data
22 0.427771578 172.217.100.37 192.168.0.136 TCP 66 443 -> 60810 [ACK] Seq=4967 Ack=795 Win=66816 Len=0 TSval=1251533316 TSecr=2414623290
23 0.427771632 172.217.100.37 192.168.0.136 TLSv1.3 97 Application Data
24 0.427794679 192.168.0.136 172.217.100.37 TCP 66 60810 -> 443 [ACK] Seq=826 Ack=4998 Win=64128 Len=0 TSval=2414623298 TSecr=1251533316
25 0.436511654 172.217.100.37 192.168.0.136 TCP 66 443 -> 60810 [ACK] Seq=4998 Ack=826 Win=66816 Len=0 TSval=1251533326 TSecr=2414623297
26 0.758131358 172.217.100.37 192.168.0.136 TLSv1.3 484 Application Data
27 0.758147677 192.168.0.136 172.217.100.37 TCP 66 60810 -> 443 [ACK] Seq=826 Ack=5416 Win=64128 Len=0 TSval=2414623629 TSecr=1251533554
28 0.758131406 172.217.100.37 192.168.0.136 TLSv1.3 481 Application Data

> Frame 21: 97 bytes on wire (776 bits), 97 bytes captured (776 bytes) on interface eth0, id 0
> Ethernet II, Src: VMWare_74:24:c3 (00:0c:29:74:24:c3), Dst: TendaTec_09:95:c0 (50:2b:73:09:95:c0)
> Internet Protocol Version 4, Src: 192.168.0.136, Dst: 172.217.100.37
> Transmission Control Protocol, Src Port: 60810, Dst Port: 443, Seq: 795, Ack: 4967, Len: 31
- Transport Layer Security
  - TLSv1.3 Record Layer: Application Data Protocol: http-over-tls
    Opaque Type: Application Data (23)
    Version: TLS 1.2 (0x0303)
    Length: 26
    Encrypted Application Data: 72b828c1a3c20944d654f095904446b680ce0da8e9d67c4b5d02
    [Application Data Protocol: http-over-tls]

0000  50 2b 73 09 95 c0 00 0c  29 74 24 c3 08 00 45 00
0010  00 53 8f f2 40 00 40 06  96 83 c0 a8 00 88 ac d9
0020  a6 25 ed 8a 01 bb 2c ba  c5 9d 6a 4b e0 a5 80 18
0030  01 f5 14 75 00 00 01  01 08 0a 8f ec 3a 41 4a 98
0040  e2 02 17 03 03 00 1a  72 b8 28 c1 a3 c2 09 44 d6
0050  54 f0 95 90 44 46 b6 80  ce 0d a8 e9 d6 7c 4b 5d
0060  02

P+s ..... )t$ .. E
S .. @ .. .
% .. , .. jK ..
.. u .. :AJ
..... F .. D ..
T .. DF .. . [K]
.
```

here sniffers like wireshark cannot decode the data. Hence some random person could read the username and password from the packets.