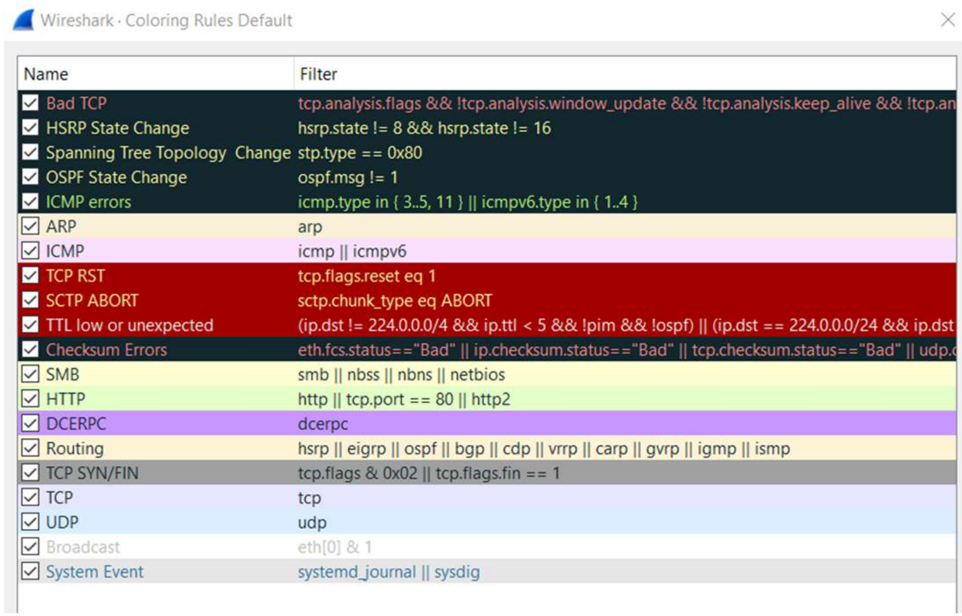
Wireshark Analysis

Group CS18:
Ridhiman Kaur Dhindsa, 210101088
Harsh Katara, 210101045
Dhruv Patel, 210101075

CS342
Assignment 2,
Question 4
Wireshark Analysis of HTTP, Web Cache, DNS, and Transport Layer Protocols

Wireshark colour coding

Color of row	Packet Type
Light purple	TCP
Light blue	UDP
Black	Packets with errors
Light green	HTTP traffic
Light yellow	Windows-specific traffic, Server Message Blocks (SMBs) etc.
Dark yellow	Routing
Grey	TCP SYN/FIN
Red	TTL low or unexpected



Wireshark filters used

- ip.addr == 172.17.1.1 or
  ip.src/ip.dst == 172.17.1.1
- tcp, udp, dns, http
- tcp.port == 80 || udp.port == 80,
  TCP or UDP port is 80 (HTTP)
- tcp contains “youtube”, udp contains “youtube”, http contains “youtube”
- frame contains “mail”

Wireshark analysis of Network packets

Activity: Watching Youtube videos

Time: 12pm Location: Hostel Internet connection: LAN

Total packets= 241266

TCP packets= 18013 = 7.5% (light purple colour)

UDP packets= 139796 = 57.9% (light blue colour)

DNS packets= 606 = 0.3% (light blue colour)

HTTP GET, POST, OK 200 packets= 48 = 0.0002% (light green colour)

ARP packets (Broadcast) = 14308 = 5.9% (light yellow colour)

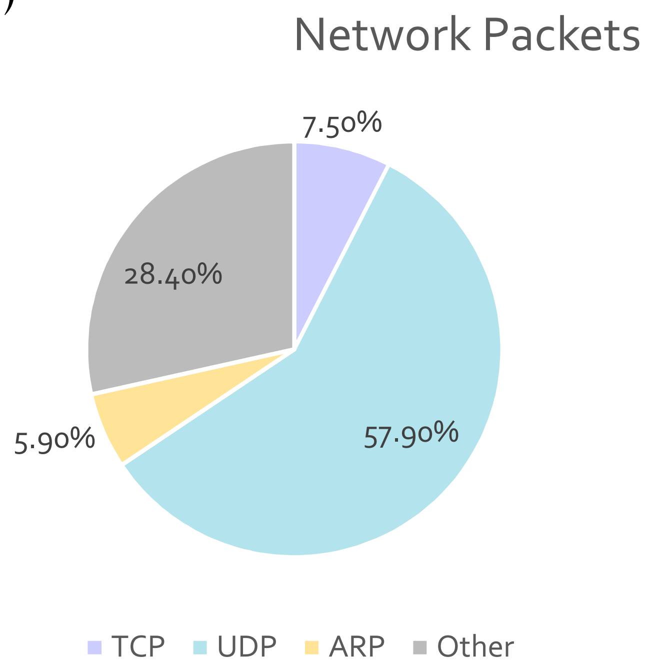
Total packets related to youtube IP address= 189+345 = 534

Packets going to youtube server= 189 = 35.4%

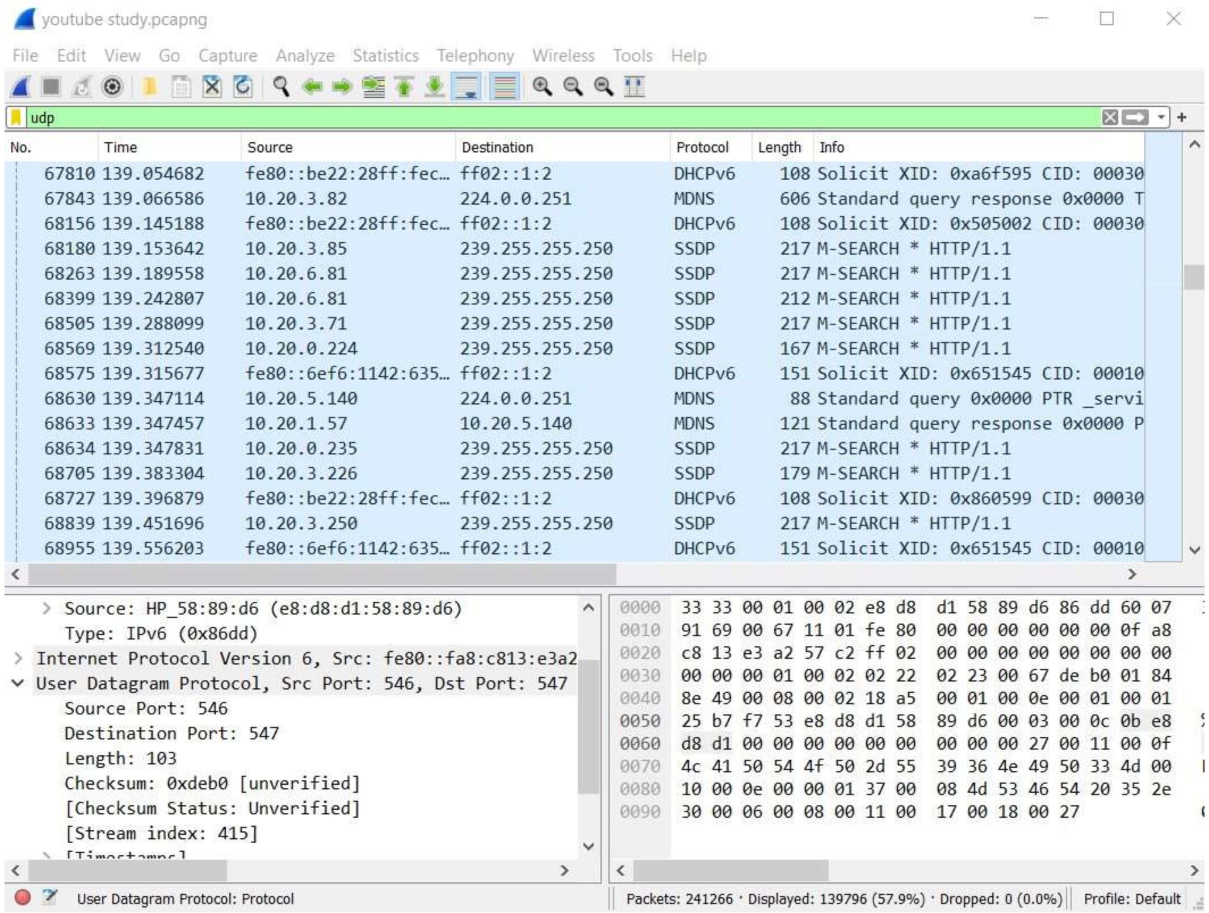
Packets coming from youtube server= 345 = 64.6%

UDP packets containing “youtube” = 25

TCP packets containing “youtube” = 0



5.9sec after starting packet capture, found 1 packet from server (172.17.1.1) to PC (10.20.1.57, port 53) which indicated a standard query response from server, i.e., the client-server handshake. The length was 548 bytes, UDP protocol, source port 53, destination port 55618. DNS response was standard- 1 question, 1 answer, no errors.



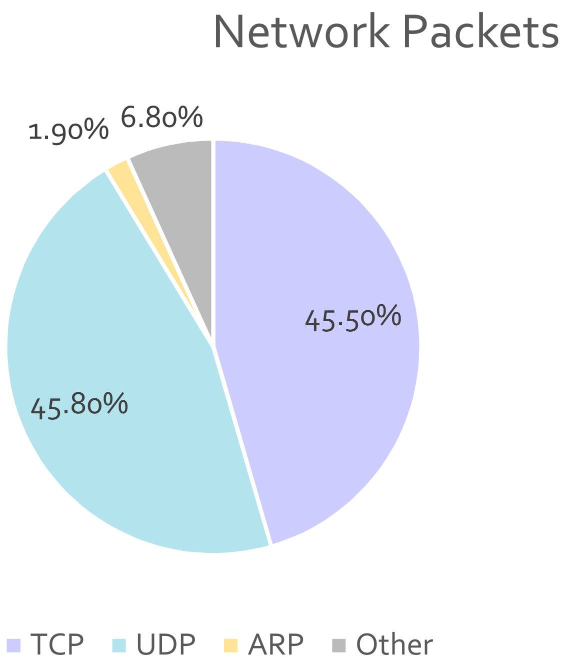
There were packets for client hello, server hello, neighbor solicitation (light pink), ARP broadcast (light yellow), router advertisement, multicast listener reports etc.

Activity: Watching Instagram reels

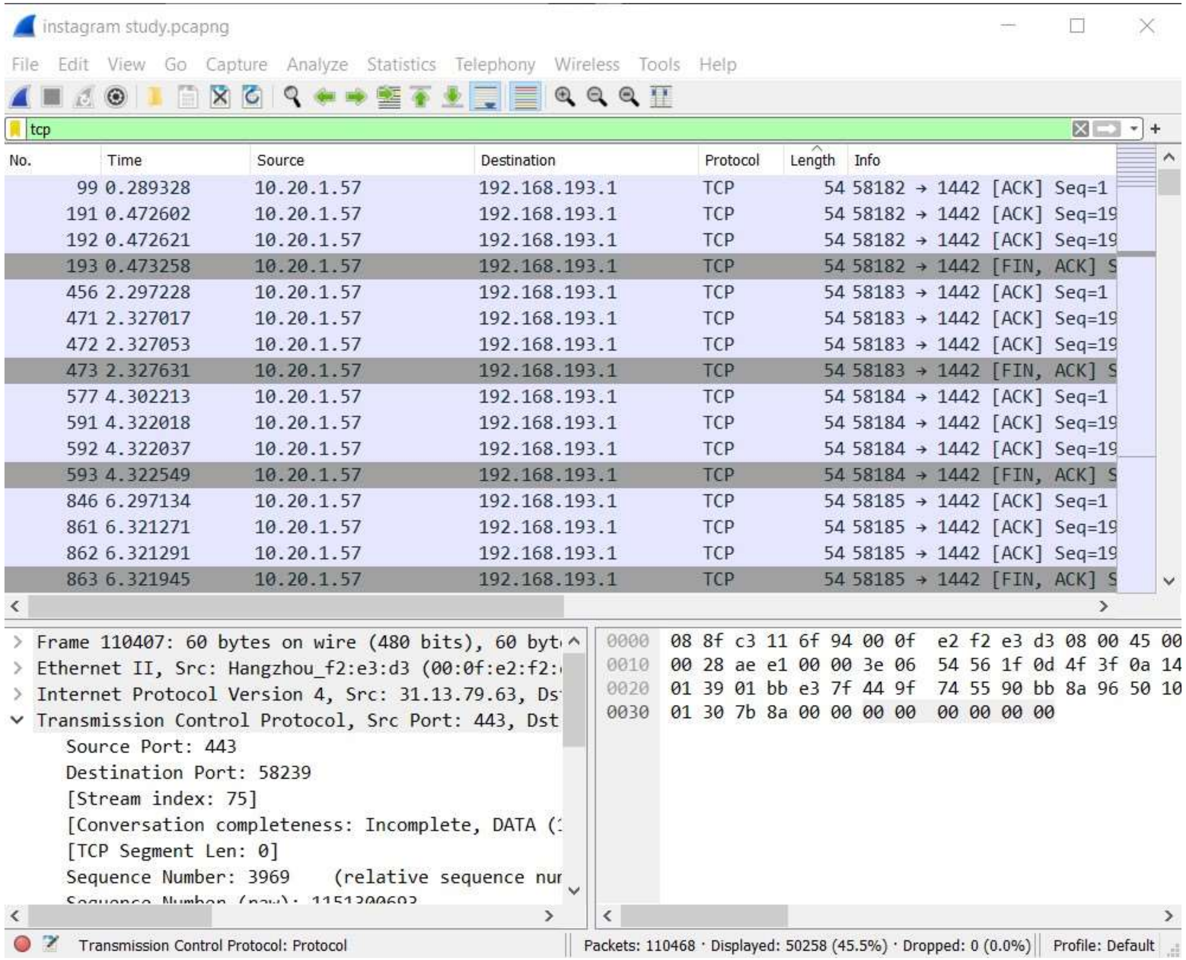
Time: 1.30pm Location: Lab Internet connection: LAN
Total packets= 110468
TCP packets= 50258 = 45.5% (light purple/grey colour)
UDP packets= 50542 = 45.8% (light blue colour)
DNS packets= 163 = 0.1% (light blue colour)
HTTP GET, POST, OK 200 packets= 6 = 0.00% (light green colour)
ARP packets (Broadcast) = 2098 = 1.9% (light yellow colour)

Total packets related to instagram IP address= 419+681 = 1100
Packets going to instagram server= 419 = 38.09%
Packets coming from instagram server= 681 = 61.90%
UDP packets containing "instagram" = 40
TCP packets containing "instagram" = 21

Most packets being sent to and from instagram servers ranged from 200-1000 bytes.



Observations:



All TCP packets related to Instagram had source IP address 10.20.1.57 and destination IP address 31.13.79.174 or 180.149.62.32. This indicates that while watching reels, most packets contain request queries from PC to Instagram servers.

It is observed that DNS packets constitute a very small percentage (0.3-0.6%) of total packets picked up by Wireshark.

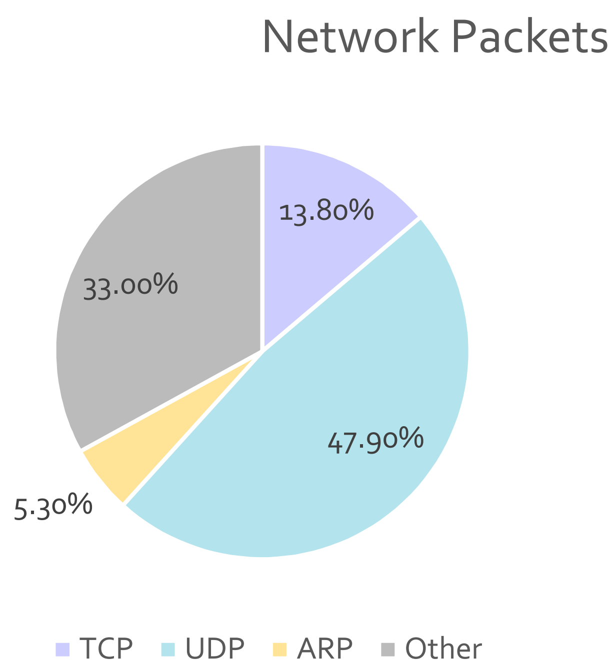
Activity: Using Gmail

Sending and receiving emails via Gmail

Time: 2.30pm Location: Lab Internet connection: LAN
Total packets= 128100
TCP packets= 17740 = 13.8% (light purple/grey colour)
UDP packets= 61374 = 47.9% (light blue colour)
DNS packets= 772 = 0.6% (light blue colour)
HTTP GET, POST, OK 200 packets= 12 = 0.00% (light green colour)
ARP packets (Broadcast) = 6764 = 5.3% (light yellow colour)

Total packets related to Gmail IP address= 17+17 = 34
Packets going to Gmail server= 17 = 50%
Packets coming from Gmail server= 17 = 50%
UDP packets containing "mail" = 34
TCP packets containing "mail" = 10

The Gmail packets have length approximately 380 bytes. Protocol= UDP, IPv4, Time to live (TTL)= 63 sec. Since it is Google (Gmail) so port 53 was used. There were 10 TCP packets with TLS Handshake protocol: Client Hello. No explicit email body could be found as Gmail uses SSL encryption for privacy and secure emailing.



Activity: Online gaming

Playing online game paper.io

Time: 3pm Location: Hostel Internet connection: Airtel

Total packets= 1331

TCP packets= 772 = 58% (light purple/grey colour)

UDP packets= 543 = 40.8% (light blue colour)

DNS packets= 84 = 6.3% (light blue colour)

HTTP GET, POST, OK 200 packets= 0 = 0.00% (light green colour)

ARP packets (Broadcast) = 16 = 1.2% (light yellow colour)

Total packets related to gaming IP address= 84

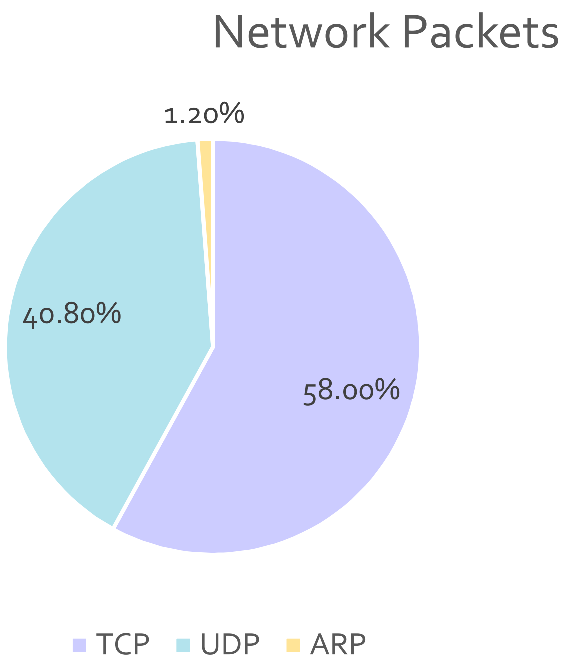
Packets going to gaming server= 42 = 50%

Packets coming from gaming server= 42 = 50%

UDP packets containing "paper.io" = 12

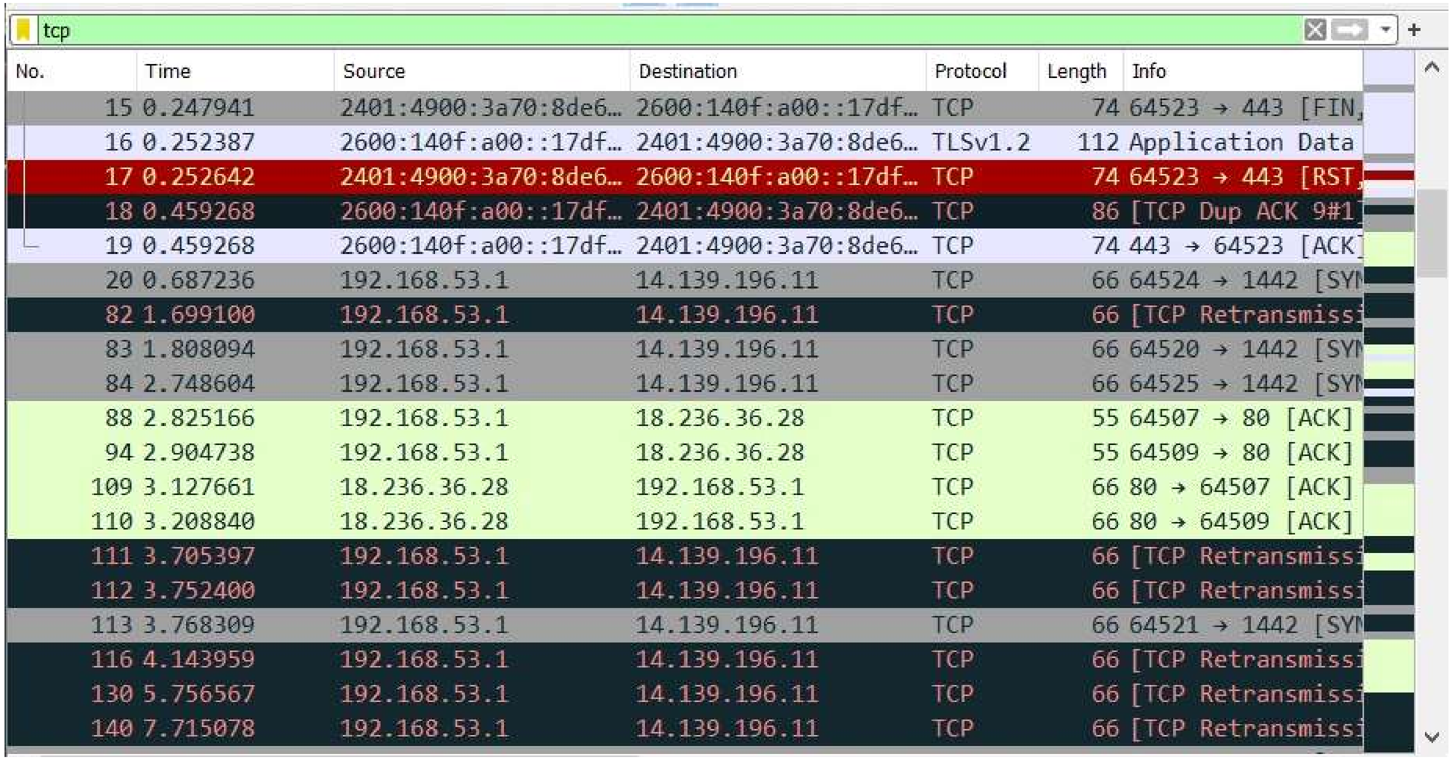
TCP packets containing "paper.io" = 0

The length of gaming packets sent over the network, range from 70-120 bytes.



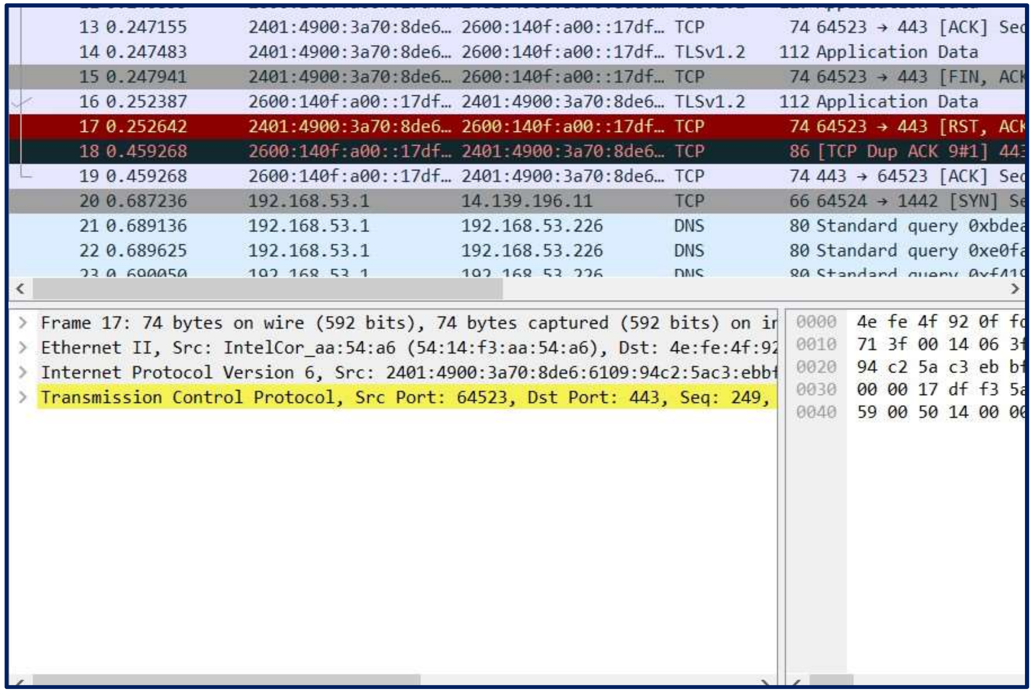
On visiting sites under HTTP protocol

Websites under HTTP protocol are not considered secure anymore, unlike those which open under HTTPS. HTTPS is essentially a more secure version of HTTP. It uses SSL/TLS to encrypt connections between web browsers and servers. This is why most packets in our previous activities contained TLS (Transport Layer Security) section.

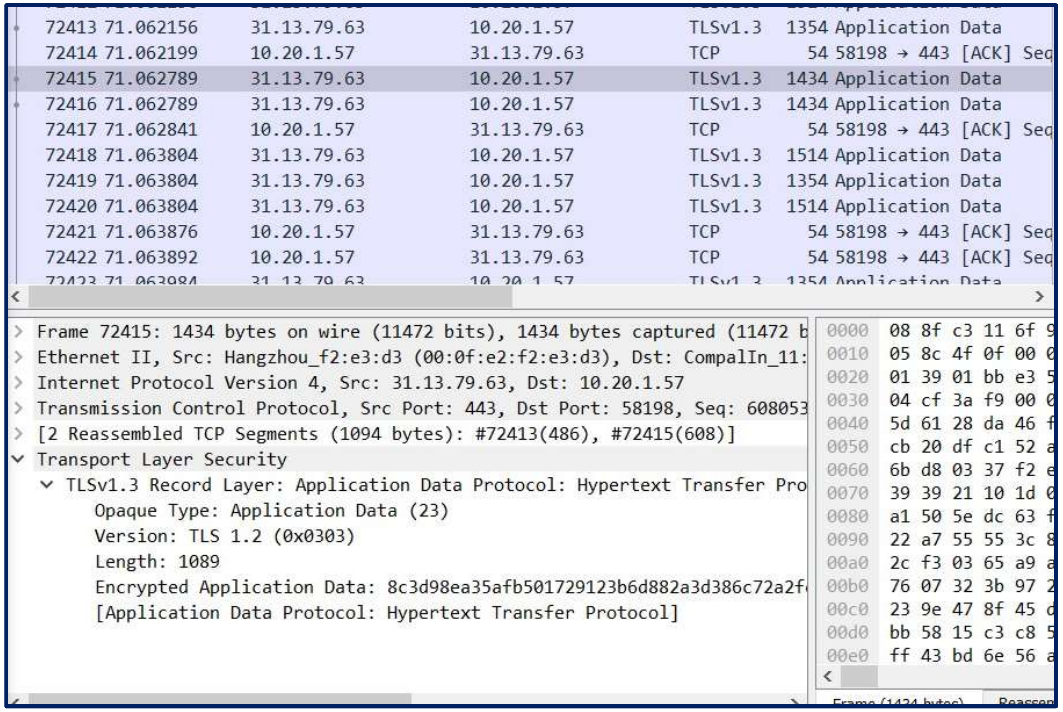

tcp

No.	Time	Source	Destination	Protocol	Length	Info
15	0.247941	2401:4900:3a70:8de6…	2600:140f:a00::17df…	TCP	74	64523 → 443 [FIN,
16	0.252387	2600:140f:a00::17df…	2401:4900:3a70:8de6…	TLSv1.2	112	Application Data
17	0.252642	2401:4900:3a70:8de6…	2600:140f:a00::17df…	TCP	74	64523 → 443 [RST,
18	0.459268	2600:140f:a00::17df…	2401:4900:3a70:8de6…	TCP	86	[TCP Dup ACK 9#1]
19	0.459268	2600:140f:a00::17df…	2401:4900:3a70:8de6…	TCP	74	443 → 64523 [ACK]
20	0.687236	192.168.53.1	14.139.196.11	TCP	66	64524 → 1442 [SYN
82	1.699100	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi
83	1.808094	192.168.53.1	14.139.196.11	TCP	66	64520 → 1442 [SYN
84	2.748604	192.168.53.1	14.139.196.11	TCP	66	64525 → 1442 [SYN
88	2.825166	192.168.53.1	18.236.36.28	TCP	55	64507 → 80 [ACK]
94	2.904738	192.168.53.1	18.236.36.28	TCP	55	64509 → 80 [ACK]
109	3.127661	18.236.36.28	192.168.53.1	TCP	66	80 → 64507 [ACK]
110	3.208840	18.236.36.28	192.168.53.1	TCP	66	80 → 64509 [ACK]
111	3.705397	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi
112	3.752400	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi
113	3.768309	192.168.53.1	14.139.196.11	TCP	66	64521 → 1442 [SYN
116	4.143959	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi
130	5.756567	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi
140	7.715078	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi

In the above picture, most rows are red or black, indicating a non-secure connection.



HTTP sites don't have TLS section



HTTPS sites have TLS section indicating the version of TLS used (TLS 1.2), length, encrypted application data and protocol (Hypertext Transfer Protocol).

Answers to assignment questions

Part 1)

Each packet typically contains the following fields:

- Frame- number, length, arrival time, epoch time, coloring rules etc.
- Ethernet- Source, Destination, Type (IPv4)
- Internet Protocol Version 4- differentiated services, TTL, protocol(TCP/UDP), src/dest address
- TCP/UDP field- src/dest port, segment length, flags, checksum etc.
- HTTP field- status code (e.g. 200 OK), response version (HTTP/1.1), date, content type, cache control
- Line based text data

The protocols involved include:

- Application Layer: HTTP, HTTPS, DNS.
- Transport Layer: This layer comprises of TCP, UDP, TLS
- Network Layer: IPv4
- Link Layer: ARP (Address Resolution Protocol), broadcasting over LAN

Contd…

Part 1) contd.

HTTP Packet format:

```
Hypertext Transfer Protocol
v HTTP/1.1 200 OK\r\n
   > [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
     Response Version: HTTP/1.1
     Status Code: 200
     [Status Code Description: OK]
     Response Phrase: OK
> Content-Length: 22\r\n
  Date: Sun, 10 Sep 2023 06:52:36 GMT\r\n
  Connection: close\r\n
  Content-Type: text/plain\r\n
  Cache-Control: max-age=30, must-revalidate\r\n
  \r\n
  [HTTP response 1/1]
  [Time since request: 0.376595000 seconds]
  [Request in frame: 83547]
  [Request URI: http://www.msftconnecttest.com/connecttest.txt]
  File Data: 22 bytes
```

Status Line: Contains the HTTP version (HTTP/1.1), a numeric status code (200 OK, 404 Not Found), a reason phrase (OK, Not Found) indicating the outcome of the request.

Headers: Key-value pairs providing additional information about the response, such as the Content-Type, Content-Length etc.

Blank Line: A blank line separating the headers from the message body (if any).

Message Body: Contains the actual content of the response, such as HTML, JSON, or binary data.

DNS packet format:

```
Domain Name System (response)
  Transaction ID: 0xa253
> Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 0
  Authority RRs: 1
  Additional RRs: 0
v Queries
   > optimizationguide-pa.googleapis.com: type HTTPS, class IN
v Authoritative nameservers
   > googleapis.com: type SOA, class IN, mname ns1.google.com
  [Request In: 65469]
  [Time: 0.001794000 seconds]
```

DNS Header (12 bytes): Identification (ID), Flags, Question Count, Answer Count, Authority Count, Additional Count- all 2 bytes each.

Question Section (Variable Length): Contains one or more DNS question records (QNAME, QTYPE, QCLASS).

Answer Section (Variable Length): Contains resource records (RRs) with answers to the DNS query.

Authority Section (Variable Length): Contains RRs that point to authoritative DNS servers for the queried domain.

Additional Information Section (Variable Length): Contains additional RRs that provide extra information about the domain in question.

Part 2) Source IP fields contain the IP address of the machine sending the request (most cases our PC) and destination IP field contains the IP address of server to which the request is directed (such as Instagram, YouTube, Google). Port number indicates the port from which request originates, e.g. Port 53 for Google related queries, port 587 for secure emailing etc. Destination port describes the port number on server where request must be entertained. Ethernet address (MAC) contains unique ID, src/dest addresses, routing information etc. Protocol number indicates TCP (6), UDP (17) etc.

Part 3)

Sequence of messages while watching Youtube Videos:

1. **DNS Resolution (Client to DNS Server):** Protocol: Typically uses DNS (Domain Name System).
2. **TCP Handshake (Client to YouTube Server):** Protocol: TCP (Transmission Control Protocol). The three-way handshake (SYN, SYN-ACK, ACK) is used to establish the connection.
3. **HTTPS/TLS Handshake (Client to YouTube Server):** Protocol: HTTPS (HTTP over TLS/SSL). The client validates the server's certificate.
4. **HTTP Request for Video (Client to YouTube Server):** Protocol: HTTP (Hypertext Transfer Protocol) or HTTPS. The client sends an HTTP GET request to request a specific video resource.
5. **HTTP Response with HTML Page (YouTube Server to Client):** Protocol: HTTP or HTTPS.
6. **Client Parses HTML and Loads Video Player (Client-Side):** Protocol: HTML and JavaScript.
7. **HTTP Request for Video Manifest (Client to YouTube Server):** Protocol: HTTP or HTTPS. The client sends an HTTP request for the video manifest, which can be in DASH or HLS format.
8. **HTTP Response with Video Manifest (YouTube Server to Client):** Protocol: HTTP or HTTPS. The YouTube server responds with the video manifest, providing information about video quality options and URLs for video segments.
9. **Client Selects Playback Quality (Client-Side):** Protocol: JavaScript.
10. **HTTP Request for Video Segments (Client to YouTube Server):** Protocol: HTTP or HTTPS.
11. **HTTP Responses with Video Segments (YouTube Server to Client):** Protocol: HTTP or HTTPS.
12. **Client Continues to Fetch Segments (Streaming) (Client-Side):** Protocol: JavaScript.

Contd…..

Part 3) Contd.

Handshaking mechanism:

- **TCP Handshake (Client to YouTube Server):** This involves a three-way handshake:

SYN (Synchronize): The client sends a TCP packet with the SYN flag set to request a connection.
SYN-ACK (Synchronize-Acknowledge): The YouTube server responds with a packet that has the SYN and ACK (Acknowledge) flags set, indicating its willingness to establish a connection.
ACK: The client acknowledges the server's response by sending a final ACK packet. At this point, the TCP connection is established.

- **HTTPS/TLS Handshake (Client to YouTube Server):** The handshake includes the following steps:

ClientHello: The client sends a message specifying its supported encryption algorithms and other parameters.
ServerHello: The YouTube server responds with its selected encryption settings.
Certificate Exchange: The server sends its digital certificate to prove its identity.
Key Exchange: The client and server exchange cryptographic keys.
Finished: Both parties confirm the handshake completion.

Part 4)

DNS resolution is the first step in accessing any website or service, including YouTube, Gmail, Instagram. It translates human-readable domain names (e.g., www.youtube.com) into IP addresses (e.g., 216.58.192.14), allowing clients to locate appropriate servers. TCP is used for the establishment of reliable, connection-oriented communication between the client's device and YouTube's servers. UDP may be used for YouTube/Instagram, but for email, TCP and SMTP must be used so that there's no loss of data. HTTPS/TLS is essential for secure communication between the client and server. It encrypts data exchanged during the handshake and subsequent requests, protecting user privacy and preventing data interception.

Part 5)
The Youtube packet data contained 139 packets with HTTP status code: 304 Not Modified. This indicates that these packets were cached on the system and weren't required to be fetched from server. This is part of **Video Segment Caching:**
Video segments, which make up the video stream, can be cached temporarily by the client-side video player. As a user watches a video, previously downloaded segments may be stored in memory or on disk for near-future playback. This local caching helps ensure smooth video streaming and minimizes re-downloads of segments that haven't changed.

Part 6) Statistics for Youtube experiment:
Time: 12pm Location: Lab Connection: IITG LAN

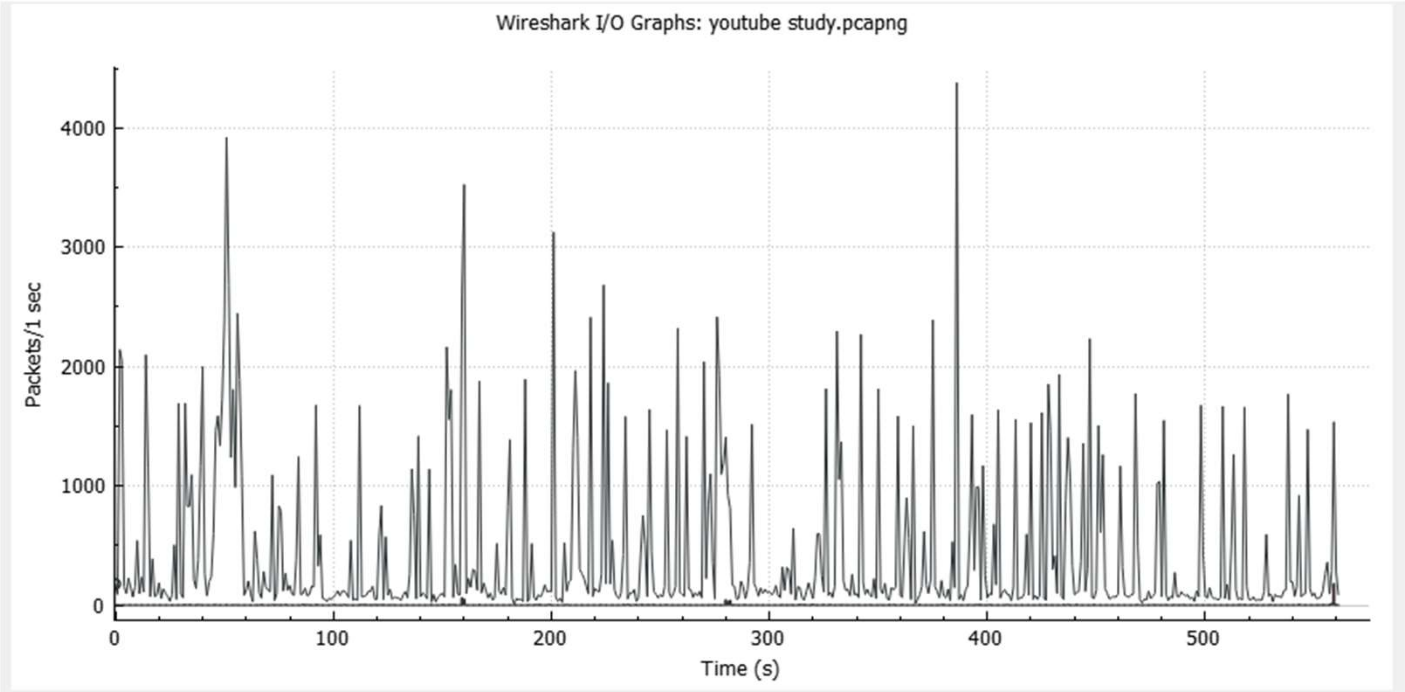
Quantity	Value
Throughput	26-27 packets/sec
RTT	17.43 ms
Packet Size	470-500 bytes
Number of packets lost	0%
TCP packets	18013 = 7.5%
UDP packets	139796 = 57.9%
Number of responses per request	1-4

Time: 5pm Location: Hostel Connection: Airtel

Quantity	Value
Throughput	22 packets/sec
RTT	19 ms
Packet Size	400-450 bytes
Number of packets lost	2%
TCP packets	6.3%
UDP packets	45.6%
Number of responses per request	1-4

Time: 10am Location: Hostel Connection: Airtel

Quantity	Value
Throughput	25 packets/sec
RTT	17.5 ms
Packet Size	400-450 bytes
Number of packets lost	0%
TCP packets	7.1%
UDP packets	47.3%
Number of responses per request	1-4


Wireshark I/O Graphs: youtube study.pcapng
Packets/1 sec
4000
3000
2000
1000
0
0
100
200
300
400
500
Time (s)

Throughput