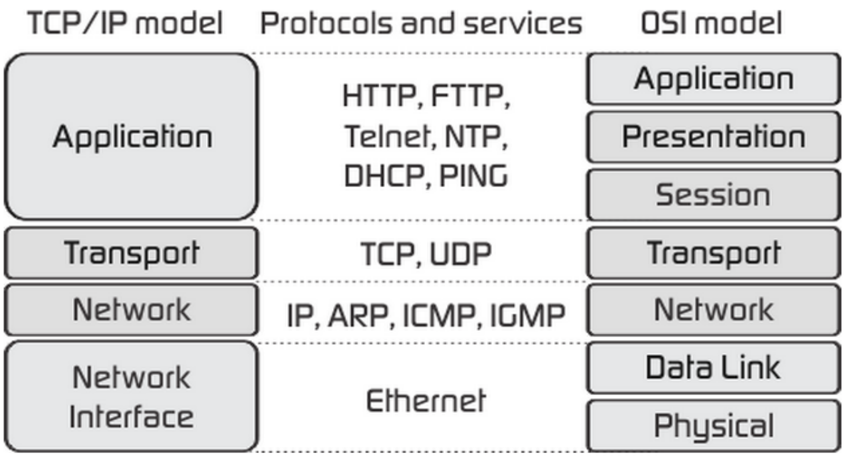
CS342 ASSIGNMENT-2

TEAM NO: CS9

Application used: Git Desktop

QUESTION-1&2:

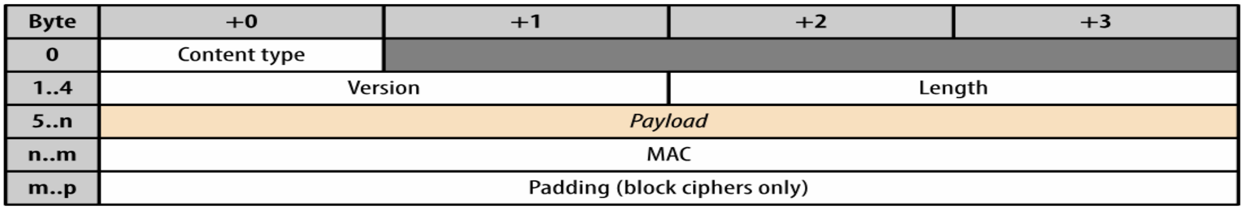
The various protocols used by GitHub Desktop applications are explained below in the respective layers that they belong to.



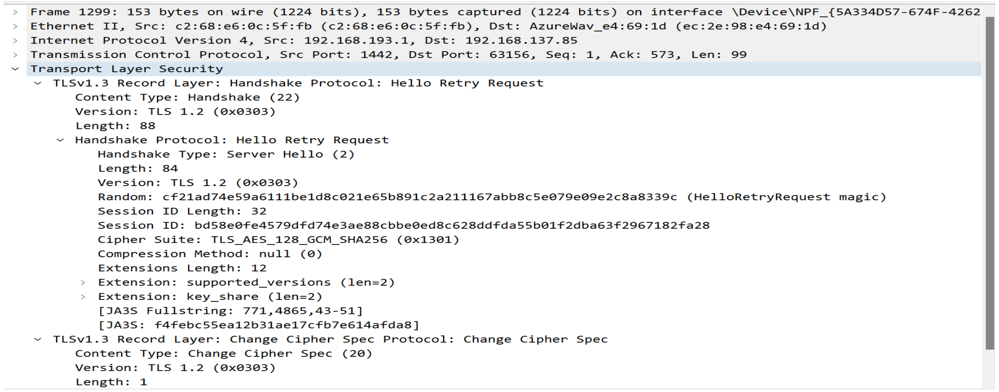
APPLICATION LAYER:

A)TLS (Transport Layer Security) Protocol

TLS secures data through encryption and message integrity. Each TLS record has a 5-byte header containing Content Type (like Handshake or App Data), Version, Length, and Payload (the actual data). Message Authentication Codes (MAC) are used for ensuring data integrity.



Byte	+0	+1	+2	+3
0	Content type			
1..4	Version		Length	
5..n	*Payload*			
n..m	MAC			
m..p	Padding (block ciphers only)			

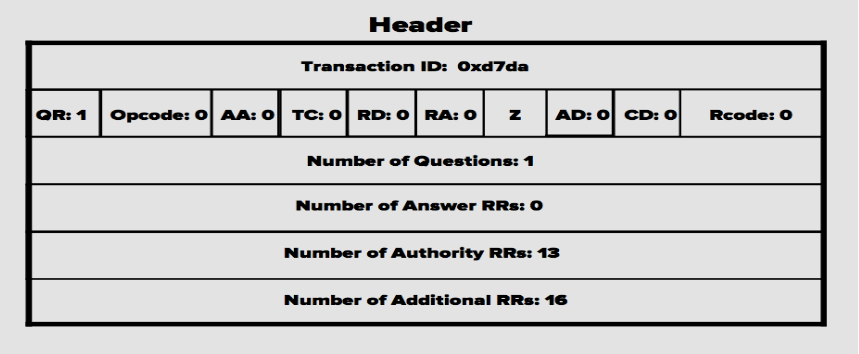


```
> Frame 1299: 153 bytes on wire (1224 bits), 153 bytes captured (1224 bits) on interface \Device\NPF_{5A334D57-674F-4262
> Ethernet II, Src: c2:68:e6:0c:5f:fb (c2:68:e6:0c:5f:fb), Dst: AzureWav_e4:69:1d (ec:2e:98:e4:69:1d)
> Internet Protocol Version 4, Src: 192.168.193.1, Dst: 192.168.137.85
> Transmission Control Protocol, Src Port: 1442, Dst Port: 63156, Seq: 1, Ack: 573, Len: 99
⌄ Transport Layer Security
  ⌄ TLSv1.3 Record Layer: Handshake Protocol: Hello Retry Request
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 88
    ⌄ Handshake Protocol: Hello Retry Request
        Handshake Type: Server Hello (2)
        Length: 84
        Version: TLS 1.2 (0x0303)
        Random: cf21ad74e59a6111be1d8c021e65b891c2a211167abb8c5e079e09e2c8a8339c (HelloRetryRequest magic)
        Session ID Length: 32
        Session ID: bd58e0fe4579dfd74e3ae88cbbe0ed8c628ddfda55b01f2dba63f2967182fa28
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Compression Method: null (0)
        Extensions Length: 12
      > Extension: supported_versions (len=2)
      > Extension: key_share (len=2)
        [JA3S Fullstring: 771,4865,43-51]
        [JA3S: f4febc55ea12b31ae17cfb7e614afda8]
  ⌄ TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
      Length: 1
```

Content Type	Handshake	It is a handshake packet.
Length	88	It is the length of the application data being transferred.
Random	cf21ad74e59a6…	32-byte pseudorandom number that is used in encryption key
Session ID	bd58e0fe457…	Used by the client to identify the session
Cipher Suite	TLS_AES_128…	List of cipher suites supported by the client

B) DNS (Domain Name system) Protocol –

DNS is a query/response protocol where clients send UDP requests with a 16-bit Transaction ID, query/response Flags, Opcode for query type, and additional info like Truncated and recursion. Questions indicate the number of requests, and responses include Answer RRs, Authority RRs, and Additional RRs to store DNS data. Queries consist of domain names and record types for resolution.



Header

Transaction ID: 0xd7da

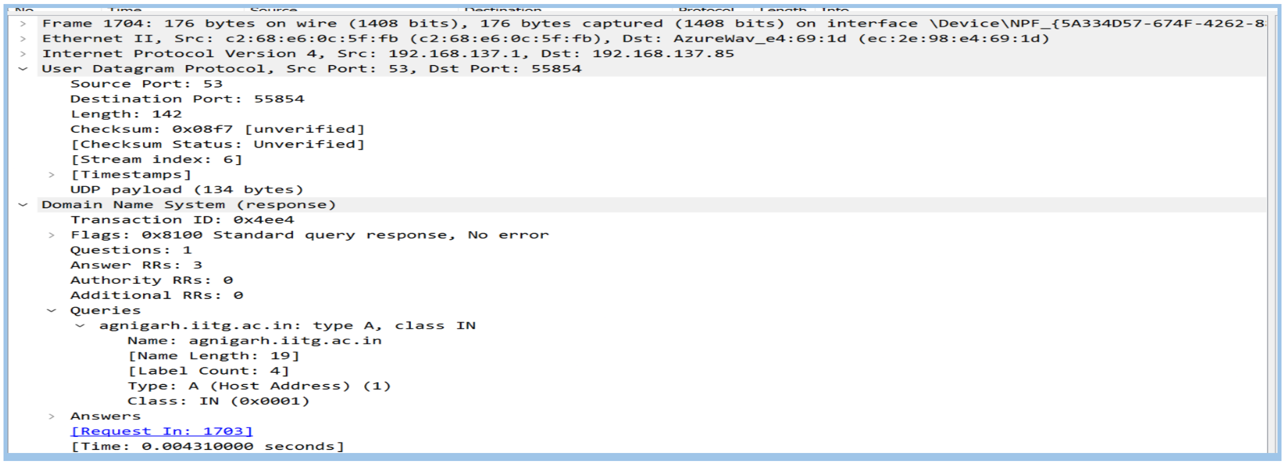
QR: 1	Opcode: 0	AA: 0	TC: 0	RD: 0	RA: 0	Z	AD: 0	CD: 0	Rcode: 0

Number of Questions: 1

Number of Answer RRs: 0

Number of Authority RRs: 13

Number of Additional RRs: 16



```
>  Frame 1704: 176 bytes on wire (1408 bits), 176 bytes captured (1408 bits) on interface \Device\NPF_{5A334D57-674F-4262-8
>  Ethernet II, Src: c2:68:e6:0c:5f:fb (c2:68:e6:0c:5f:fb), Dst: AzureWav_e4:69:1d (ec:2e:98:e4:69:1d)
>  Internet Protocol Version 4, Src: 192.168.137.1, Dst: 192.168.137.85
v  User Datagram Protocol, Src Port: 53, Dst Port: 55854
       Source Port: 53
       Destination Port: 55854
       Length: 142
       Checksum: 0x08f7 [unverified]
       [Checksum Status: Unverified]
       [Stream index: 6]
     > [Timestamps]
       UDP payload (134 bytes)
v  Domain Name System (response)
       Transaction ID: 0x4ee4
     > Flags: 0x8100 Standard query response, No error
       Questions: 1
       Answer RRs: 3
       Authority RRs: 0
       Additional RRs: 0
     v Queries
         v agnigarh.iitg.ac.in: type A, class IN
               Name: agnigarh.iitg.ac.in
               [Name Length: 19]
               [Label Count: 4]
               Type: A (Host Address) (1)
               Class: IN (0x0001)
     > Answers
       [Request In: 1703]
       [Time: 0.004310000 seconds]
```

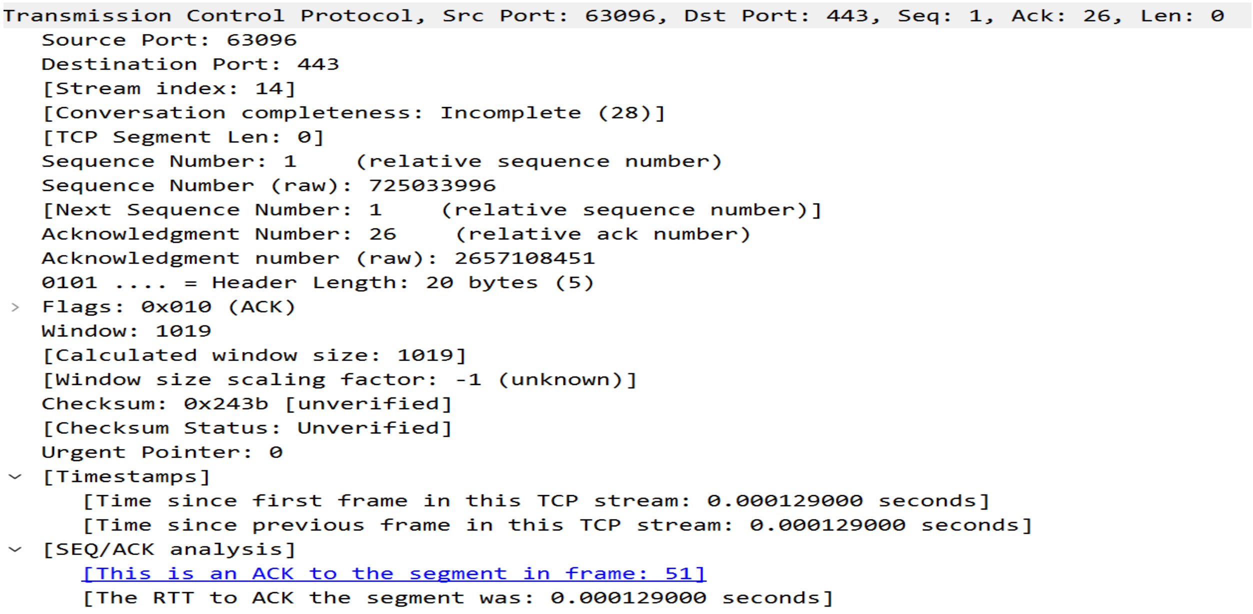
WIRESHARK ANALYSIS

Transaction ID	0x4ee4	It is a handshake packet.
Flags	0x8100 Standard query response	Message is response for a query, and it is a standard query
Questions	1	1 request received in DNS query segment
Answer RRs	3	In the segment there are 3 resource records
Authority RRs/ Additional RRs	0	No authority resource records and additional resources records received here
Name	anigarh.iitg.ac.in	Name of the server
Type	A	Response for the IPv4 address of the server

TRANSPORT LAYER:

A) TCP (Transmission Control Protocol) –

TCP is a fundamental networking standard that outlines the procedures for initiating and sustaining a network dialogue, enabling application programs to share data. As a connection-oriented protocol, TCP establishes a link before data exchange begins among devices. TCP stands as the prevalent protocol in networks reliant on the Internet Protocol (IP), and the conjunction of TCP and IP is occasionally denoted as TCP/IP.



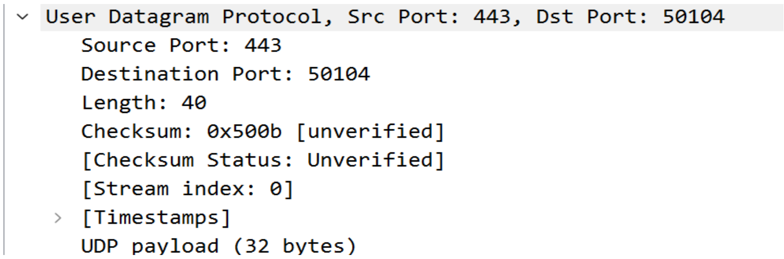
```
Transmission Control Protocol, Src Port: 63096, Dst Port: 443, Seq: 1, Ack: 26, Len: 0
   Source Port: 63096
   Destination Port: 443
   [Stream index: 14]
   [Conversation completeness: Incomplete (28)]
   [TCP Segment Len: 0]
   Sequence Number: 1    (relative sequence number)
   Sequence Number (raw): 725033996
   [Next Sequence Number: 1    (relative sequence number)]
   Acknowledgment Number: 26    (relative ack number)
   Acknowledgment number (raw): 2657108451
   0101 .... = Header Length: 20 bytes (5)
 > Flags: 0x010 (ACK)
   Window: 1019
   [Calculated window size: 1019]
   [Window size scaling factor: -1 (unknown)]
   Checksum: 0x243b [unverified]
   [Checksum Status: Unverified]
   Urgent Pointer: 0
 ˅ [Timestamps]
      [Time since first frame in this TCP stream: 0.000129000 seconds]
      [Time since previous frame in this TCP stream: 0.000129000 seconds]
 ˅ [SEQ/ACK analysis]
      [This is an ACK to the segment in frame: 51]
      [The RTT to ACK the segment was: 0.000129000 seconds]
```

WIRESHARK ANALYSIS

Source Port	63096	IThe source port number is used by the sending host to help keep track of new incoming connections and existing data streams.
Destination Port	443	Similar to the source port, the destination port is used by the receiver to keep track of new incoming connections.
Sequence number	1	The number assigned to the packet relative to the advent of the TCP connection.
Acknowledgement number	26	It is the sequence number of the next byte the receiver expects to receive
Urgent Pointer	0	It is used to point to data that is urgently required. Here there is no such requirement and so its value is set to 0.

B) UDP (User Datagram Protocol)–

UDP, one of the most basic communication protocols in the TCP/IP suite, employs minimal communication mechanisms. It's often regarded as an unreliable transport protocol, yet it relies on IP services to offer a best-effort delivery mechanism.



```
⌄ User Datagram Protocol, Src Port: 443, Dst Port: 50104
     Source Port: 443
     Destination Port: 50104
     Length: 40
     Checksum: 0x500b [unverified]
     [Checksum Status: Unverified]
     [Stream index: 0]
   › [Timestamps]
     UDP payload (32 bytes)
```

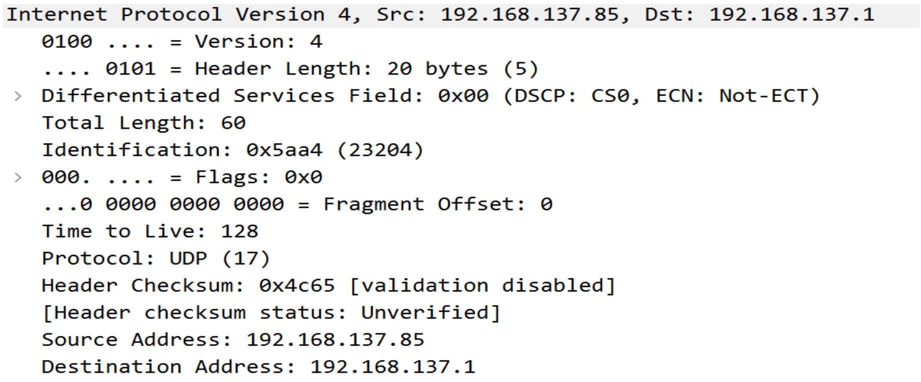
WIRESHARK ANALYSIS

Source Port	53	It is a 16-bit field and identifies the port of the sender application.
Destination Port	50104	It identifies the port of receiver application..
Length	40	It identifies the combined length of UDP Header and Encapsulated data.
Checksum	0x500b	It is calculated on UDP Header, encapsulated data and IP pseudo-header and used for error control.

NETWORK LAYER

A) IPv4 (Internet Protocol version 4)–

IPv4, the fourth iteration of the Internet Protocol (IP), stands as a foundational element of standard internetworking procedures in the global Internet and various packet-switched networks.



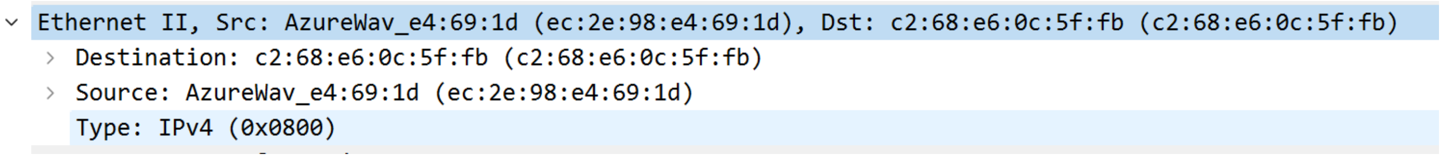
```
 Internet Protocol Version 4, Src: 192.168.137.85, Dst: 192.168.137.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x5aa4 (23204)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x4c65 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.137.85
    Destination Address: 192.168.137.1
```

WIRESHARK ANALYSIS

Version	4	Indicates the IP version used.
Header Length	20 bytes (5)	Contains the length of the IP header.
Source	192.168.137.85	The IP address of the sender
Destination	192.168.137.1	The IP address of the receiver
Time To live	128	It indicates the maximum number of hops a datagram can take to reach the destination.

LINK LAYER

A)Ethernet II: It operates as a data link layer protocol data unit and relies on the underlying Ethernet physical layer for transport. To put it differently, it encapsulates an entire Ethernet frame within its data payload when transmitted over an Ethernet link.



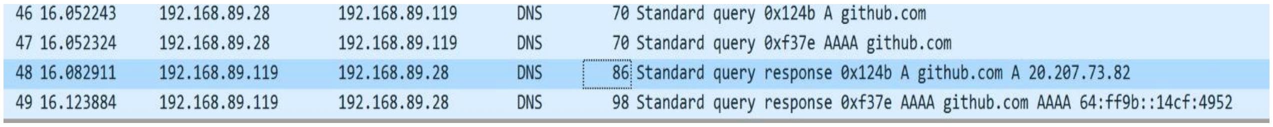
```
v Ethernet II, Src: AzureWav_e4:69:1d (ec:2e:98:e4:69:1d), Dst: c2:68:e6:0c:5f:fb (c2:68:e6:0c:5f:fb)
   > Destination: c2:68:e6:0c:5f:fb (c2:68:e6:0c:5f:fb)
   > Source: AzureWav_e4:69:1d (ec:2e:98:e4:69:1d)
     Type: IPv4 (0x0800)
```

WIRESHARK ANALYSIS

Destination	c2:68:e6:0c:5f:fb	Refers to the MAC address of the destination server
Source	ec:2e:98:e4:69:1d	Refers to the MAC address of the source server
Type	IPv4(0x0800)	Means the upper layer protocol used is IPv4

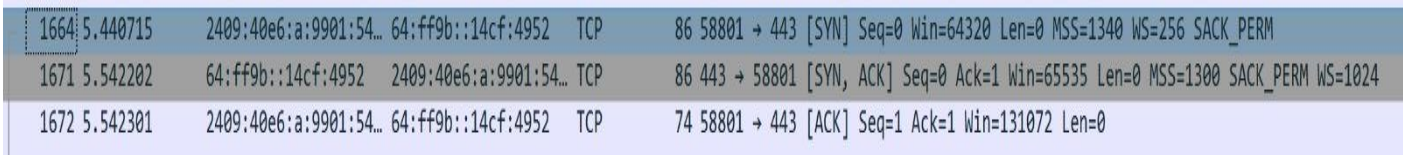
QUESTION 3

- **DNS Query:** The first step in any operation is to find the IP address of the GitHub server by performing a DNS query. The system's initial point of reference is its local DNS cache. If the domain name and its associated IP address were recently queried and are still within their time-to-live (TTL) period, the system can rely on the cached information. This helps save time and network resources, and it's a practice often observed when accessing regularly visited websites. In my case, my computer retrieved the data from the local DNS server cache.



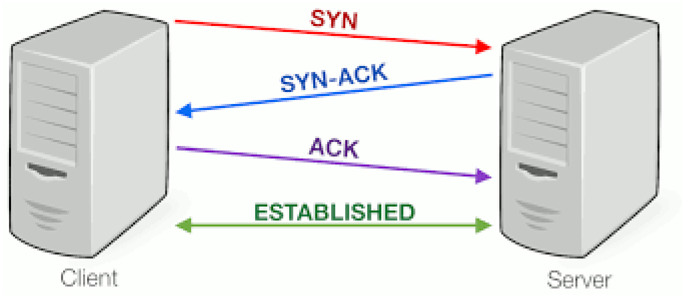
```
46 16.052243   192.168.89.28    192.168.89.119   DNS   70 Standard query 0x124b A github.com
47 16.052324   192.168.89.28    192.168.89.119   DNS   70 Standard query 0xf37e AAAA github.com
48 16.082911   192.168.89.119   192.168.89.28    DNS   86 Standard query response 0x124b A github.com A 20.207.73.82
49 16.123884   192.168.89.119   192.168.89.28    DNS   98 Standard query response 0xf37e AAAA github.com AAAA 64:ff9b::14cf:4952
```

- **TCP 3-Way Handshake:** A connection between the client and the server is established in 3 steps. As we can see in the image below, the connection is established using client port 58801 and server port 443.

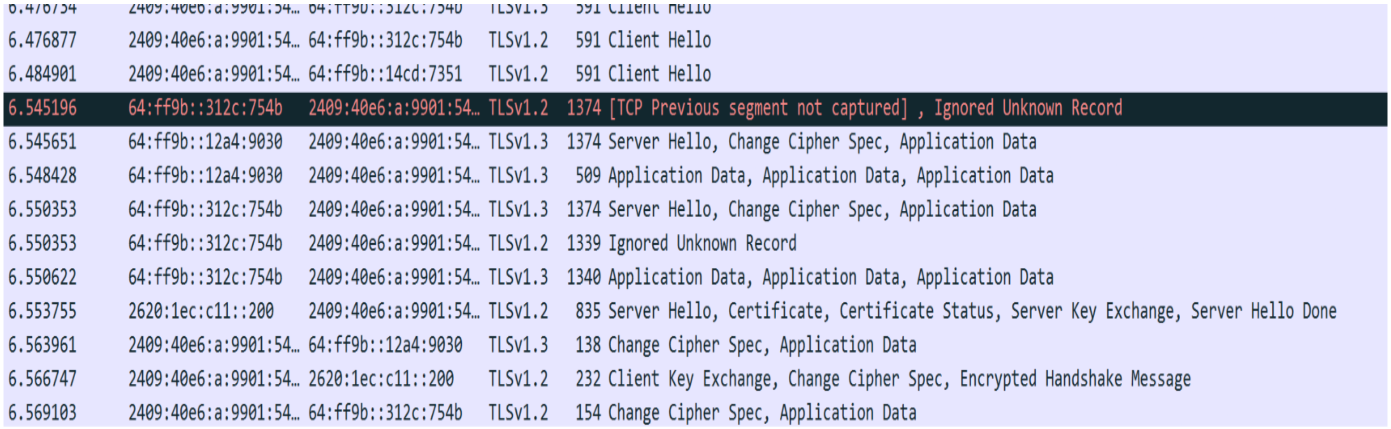


```
1664 5.440715   2409:40e6:a:9901:54…  64:ff9b::14cf:4952    TCP   86 58801 → 443 [SYN] Seq=0 Win=64320 Len=0 MSS=1340 WS=256 SACK_PERM
1671 5.542202   64:ff9b::14cf:4952    2409:40e6:a:9901:54…  TCP   86 443 → 58801 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1300 SACK_PERM WS=1024
1672 5.542301   2409:40e6:a:9901:54…  64:ff9b::14cf:4952    TCP   74 58801 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
```

The connection starts with the client and server picking initial sequence numbers, exchanged in SYN and SYN/ACK packets. They acknowledge each other's sequence numbers by incrementing them, known as the acknowledgment number. This helps detect missing or out-of-order data segments. After connection, ACKs follow for each segment, and it ends with an RST (reset) or FIN (graceful closure).



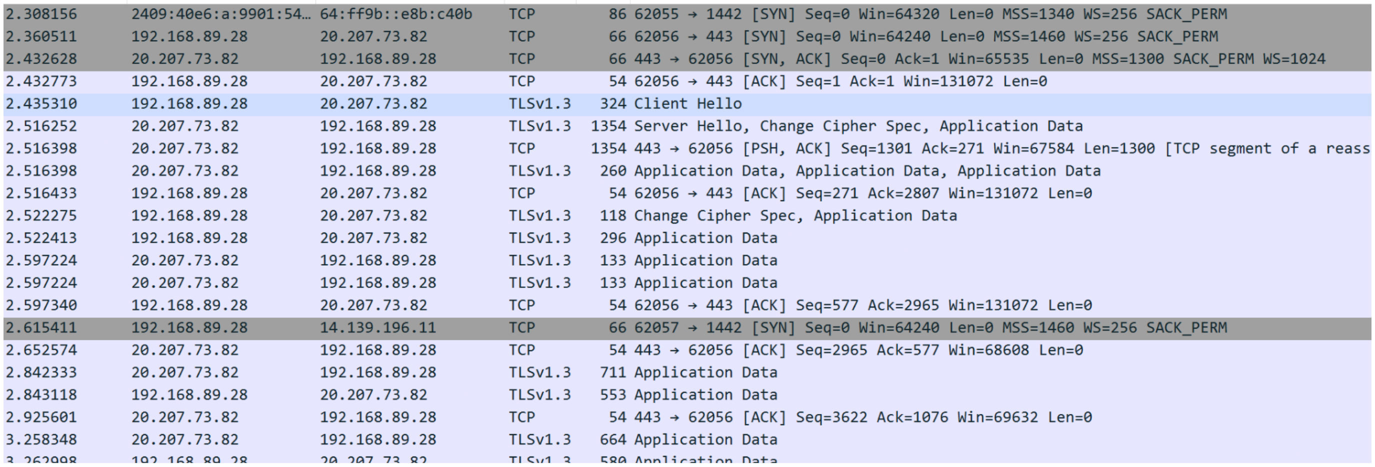
- **TLS Handshake:**The TLS handshake initiates a secure communication session using TLS encryption. It involves message exchange to acknowledge and verify each party, establish encryption methods, and agree on session keys. The process starts with the Client Hello message from the client. The server responds with the Server Hello, Server Certificate for authentication, and a Server Key. The Server Hello Done signals the server's readiness for the client's response. The client sends the Client Key and receives a New Session Ticket. This establishes the TLS session, allowing secure application data exchange between the server and client.



```
6.476734    2409:40e6:a:9901:54… 64:ff9b::312c:754b  TLSv1.3  591 Client Hello
6.476877    2409:40e6:a:9901:54… 64:ff9b::312c:754b  TLSv1.2  591 Client Hello
6.484901    2409:40e6:a:9901:54… 64:ff9b::14cd:7351  TLSv1.2  591 Client Hello
6.545196    64:ff9b::312c:754b   2409:40e6:a:9901:54… TLSv1.2 1374 [TCP Previous segment not captured] , Ignored Unknown Record
6.545651    64:ff9b::12a4:9030   2409:40e6:a:9901:54… TLSv1.3 1374 Server Hello, Change Cipher Spec, Application Data
6.548428    64:ff9b::12a4:9030   2409:40e6:a:9901:54… TLSv1.3  509 Application Data, Application Data, Application Data
6.550353    64:ff9b::312c:754b   2409:40e6:a:9901:54… TLSv1.3 1374 Server Hello, Change Cipher Spec, Application Data
6.550353    64:ff9b::312c:754b   2409:40e6:a:9901:54… TLSv1.2 1339 Ignored Unknown Record
6.550622    64:ff9b::312c:754b   2409:40e6:a:9901:54… TLSv1.3 1340 Application Data, Application Data, Application Data
6.553755    2620:1ec:c11::200    2409:40e6:a:9901:54… TLSv1.2  835 Server Hello, Certificate, Certificate Status, Server Key Exchange, Server Hello Done
6.563961    2409:40e6:a:9901:54… 64:ff9b::12a4:9030  TLSv1.3  138 Change Cipher Spec, Application Data
6.566747    2409:40e6:a:9901:54… 2620:1ec:c11::200   TLSv1.2  232 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
6.569103    2409:40e6:a:9901:54… 64:ff9b::312c:754b  TLSv1.2  154 Change Cipher Spec, Application Data
```

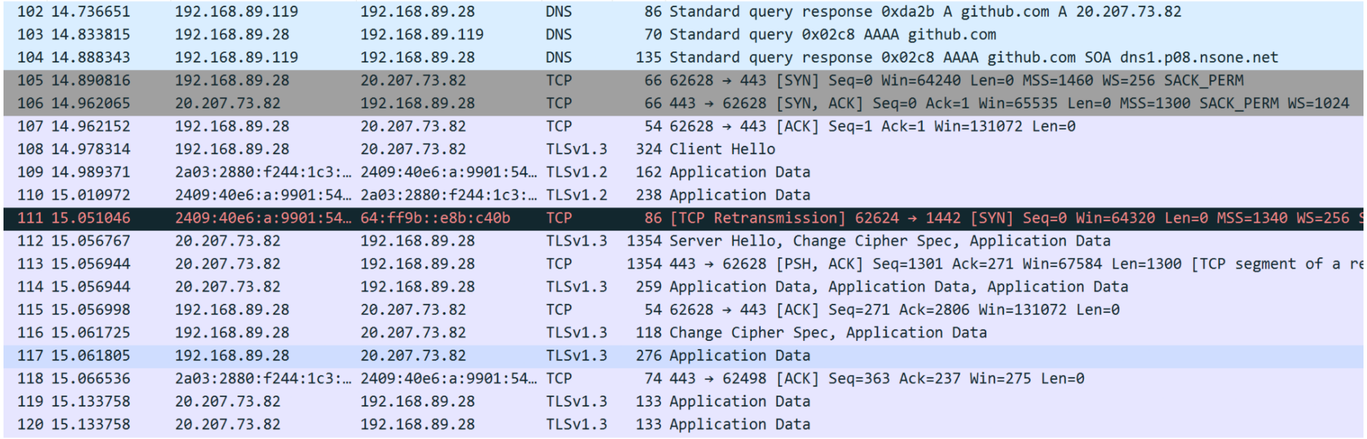
- **Sending of Resources:** After the TLS handshake, the client requests the remote repository data from the server. The server responds by sending the repository files, branches, commits, and other related data

Cloning a repository from the internet



```
2.308156   2409:40e6:a:9901:54… 64:ff9b::e8b:c40b   TCP        86 62055 → 1442 [SYN] Seq=0 Win=64320 Len=0 MSS=1340 WS=256 SACK_PERM
2.360511   192.168.89.28        20.207.73.82        TCP        66 62056 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2.432628   20.207.73.82         192.168.89.28       TCP        66 443 → 62056 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1300 SACK_PERM WS=1024
2.432773   192.168.89.28        20.207.73.82        TCP        54 62056 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
2.435310   192.168.89.28        20.207.73.82        TLSv1.3   324 Client Hello
2.516252   20.207.73.82         192.168.89.28       TLSv1.3  1354 Server Hello, Change Cipher Spec, Application Data
2.516398   20.207.73.82         192.168.89.28       TCP      1354 443 → 62056 [PSH, ACK] Seq=1301 Ack=271 Win=67584 Len=1300 [TCP segment of a reass
2.516398   20.207.73.82         192.168.89.28       TLSv1.3   260 Application Data, Application Data, Application Data
2.516433   192.168.89.28        20.207.73.82        TCP        54 62056 → 443 [ACK] Seq=271 Ack=2807 Win=131072 Len=0
2.522275   192.168.89.28        20.207.73.82        TLSv1.3   118 Change Cipher Spec, Application Data
2.522413   192.168.89.28        20.207.73.82        TLSv1.3   296 Application Data
2.597224   20.207.73.82         192.168.89.28       TLSv1.3   133 Application Data
2.597224   20.207.73.82         192.168.89.28       TLSv1.3   133 Application Data
2.597340   192.168.89.28        20.207.73.82        TCP        54 62056 → 443 [ACK] Seq=577 Ack=2965 Win=131072 Len=0
2.615411   192.168.89.28        14.139.196.11       TCP        66 62057 → 1442 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2.652574   20.207.73.82         192.168.89.28       TCP        54 443 → 62056 [ACK] Seq=2965 Ack=577 Win=68608 Len=0
2.842333   20.207.73.82         192.168.89.28       TLSv1.3   711 Application Data
2.843118   192.168.89.28        20.207.73.82        TLSv1.3   553 Application Data
2.925601   20.207.73.82         192.168.89.28       TCP        54 443 → 62056 [ACK] Seq=3622 Ack=1076 Win=69632 Len=0
3.258348   20.207.73.82         192.168.89.28       TLSv1.3   664 Application Data
3.262998   192.168.89.28        20.207.73.82        TLSv1.3   580 Application Data
```

Adding a local repository to GitHub:

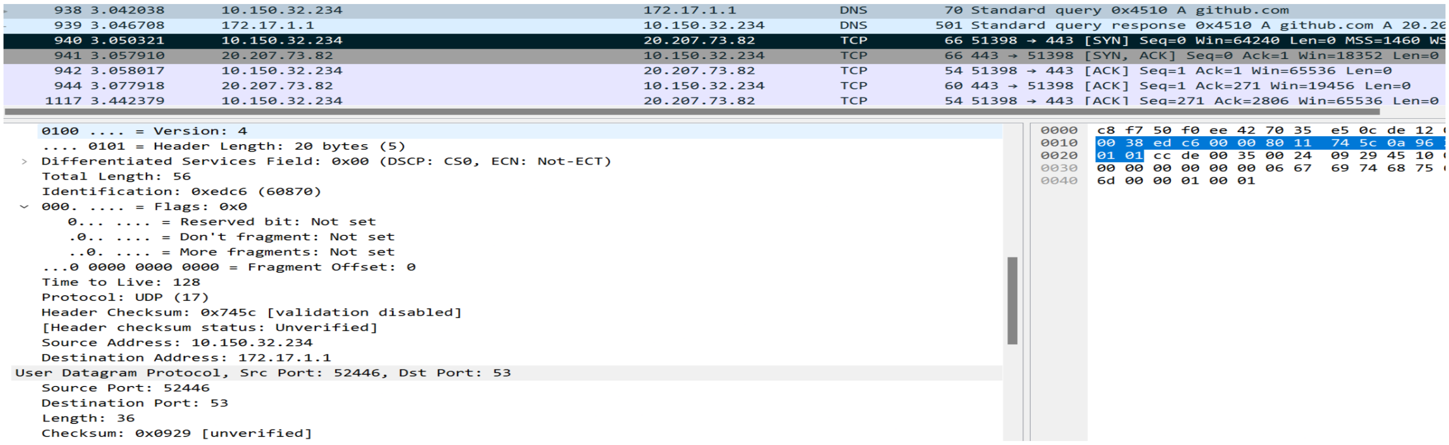


```
102 14.736651   192.168.89.119      192.168.89.28       DNS       86 Standard query response 0xda2b A github.com A 20.207.73.82
103 14.833815   192.168.89.28       192.168.89.119      DNS       70 Standard query 0x02c8 AAAA github.com
104 14.888343   192.168.89.119      192.168.89.28       DNS      135 Standard query response 0x02c8 AAAA github.com SOA dns1.p08.nsone.net
105 14.890816   192.168.89.28       20.207.73.82        TCP       66 62628 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
106 14.962065   20.207.73.82        192.168.89.28       TCP       66 443 → 62628 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1300 SACK_PERM WS=1024
107 14.962152   192.168.89.28       20.207.73.82        TCP       54 62628 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
108 14.978314   192.168.89.28       20.207.73.82        TLSv1.3  324 Client Hello
109 14.989371   2a03:2880:f244:1c3:… 2409:40e6:a:9901:54… TLSv1.2  162 Application Data
110 15.010972   2409:40e6:a:9901:54… 2a03:2880:f244:1c3:… TLSv1.2  238 Application Data
111 15.051046   2409:40e6:a:9901:54… 64:ff9b::e8b:c40b   TCP       86 [TCP Retransmission] 62624 → 1442 [SYN] Seq=0 Win=64320 Len=0 MSS=1340 WS=256 S
112 15.056767   20.207.73.82        192.168.89.28       TLSv1.3 1354 Server Hello, Change Cipher Spec, Application Data
113 15.056944   20.207.73.82        192.168.89.28       TCP     1354 443 → 62628 [PSH, ACK] Seq=1301 Ack=271 Win=67584 Len=1300 [TCP segment of a re
114 15.056944   20.207.73.82        192.168.89.28       TLSv1.3  259 Application Data, Application Data, Application Data
115 15.056998   192.168.89.28       20.207.73.82        TCP       54 62628 → 443 [ACK] Seq=271 Ack=2806 Win=131072 Len=0
116 15.061725   192.168.89.28       20.207.73.82        TLSv1.3  118 Change Cipher Spec, Application Data
117 15.061805   192.168.89.28       20.207.73.82        TLSv1.3  276 Application Data
118 15.066536   2a03:2880:f244:1c3:… 2409:40e6:a:9901:54… TCP       74 443 → 62498 [ACK] Seq=363 Ack=237 Win=275 Len=0
119 15.133758   20.207.73.82        192.168.89.28       TLSv1.3  133 Application Data
120 15.133758   20.207.73.82        192.168.89.28       TLSv1.3  133 Application Data
```

QUESTION 4

- The application's functionality heavily relies on specific protocols to operate effectively. In each case:
- **Cloning a Repository from the Internet**: To clone a repository from the internet (e.g., from GitHub), DNS is crucial for resolving the GitHub server's IP address. This information allows the client to establish a connection. Additionally, TLS (Transport Layer Security) ensures a secure and encrypted data transfer, while IPv4 facilitates network routing, and TCP guarantees reliable data transmission.
- **Adding a Local Repository to GitHub**: DNS assists in locating the GitHub server, TLS secures the data exchange during authentication and repository upload, IPv4 handles network routing, and TCP ensures the data arrives reliably.
- **Pushing a Repo into GitHub Server**:Similar to the previous functionalities, DNS is pivotal for resolving GitHub's IP address. TLS provides security, IPv4 manages network routing, and TCP guarantees the reliability of data transfer, which is crucial for pushing code changes to a remote repository.
- **Pulling a Repo from GitHub Server**:DNS plays a critical role in determining the server's IP address, enabling the client to retrieve repository data. TLS secures this data transfer, IPv4 ensures proper network routing, and TCP ensures the data is received correctly.
- **Branching a Repository**: When branching a repository, DNS helps locate the GitHub server, TLS secures data transmission during branch creation, IPv4 handles network routing, and TCP ensures the integrity of branch-related data being sent back and forth.

QUESTION 5:



```
 938 3.042038   10.150.32.234        172.17.1.1       DNS     70 Standard query 0x4510 A github.com
 939 3.046708   172.17.1.1           10.150.32.234    DNS    501 Standard query response 0x4510 A github.com A 20.20
 940 3.050321   10.150.32.234        20.207.73.82     TCP     66 51398 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS
 941 3.057910   20.207.73.82         10.150.32.234    TCP     66 443 → 51398 [SYN, ACK] Seq=0 Ack=1 Win=18352 Len=0
 942 3.058017   10.150.32.234        20.207.73.82     TCP     54 51398 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
 944 3.077918   20.207.73.82         10.150.32.234    TCP     60 443 → 51398 [ACK] Seq=1 Ack=271 Win=19456 Len=0
1117 3.442379   10.150.32.234        20.207.73.82     TCP     54 51398 → 443 [ACK] Seq=271 Ack=2806 Win=65536 Len=0
```

```
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0xedc6 (60870)
  ˅ 000. .... = Flags: 0x0
       0... .... = Reserved bit: Not set
       .0.. .... = Don't fragment: Not set
       ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x745c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.150.32.234
    Destination Address: 172.17.1.1
 User Datagram Protocol, Src Port: 52446, Dst Port: 53
    Source Port: 52446
    Destination Port: 53
    Length: 36
    Checksum: 0x0929 [unverified]
```

```
0000  c8 f7 50 f0 ee 42 70 35  e5 0c de 12 
0010  00 38 ed c6 00 00 80 11  74 5c 0a 96 
0020  01 01 cc de 00 35 00 24  09 29 45 10 
0030  00 00 00 00 00 00 06 67  69 74 68 75 
0040  6d 00 00 01 00 01
```

- When inspecting captured DNS packets, take note of the "Time to Live" (TTL) value within the DNS IPv4 header of response packets. This TTL value serves as a measure of how long the DNS data can be cached. If the TTL is greater than zero, it signifies that caching is active and the data can be stored for the specified duration.

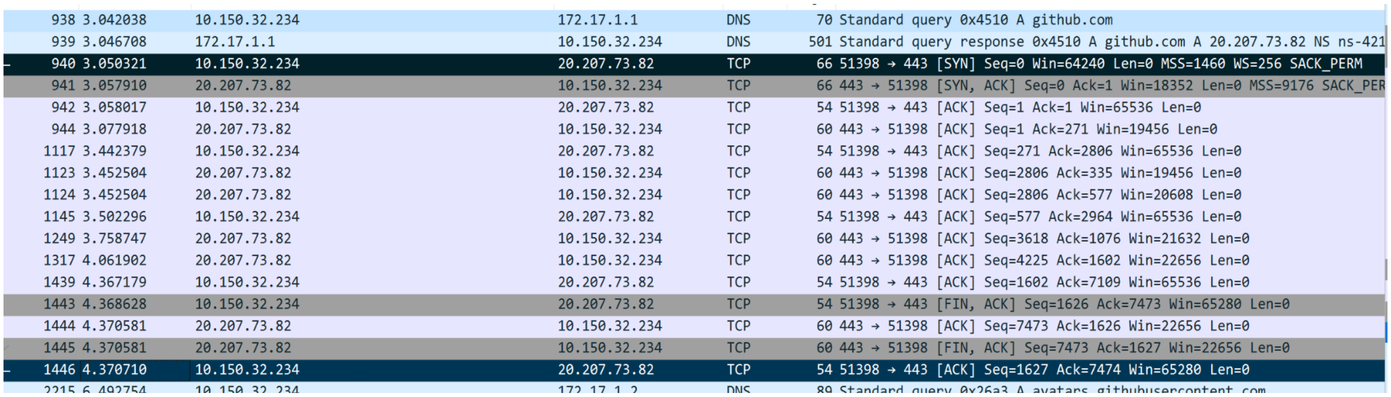
QUESTION 6:

I conducted the cloning procedure at three distinct times during the day. The resulting data is presented in the table below and can be cross-checked using the provided trace files. The communication between the client and server involves both TCP and TLS packets, which were taken into account when determining the throughput, round-trip time (RTT), average packet size, and number of responses per request.

Time	Throughput (bytes/sec)	RTT(ms)	Avg Packet Size(bytes)	Packet Lost	UDP Packets	TCP Packets and TLS packets	Number of responses per request (avg)
11 AM(lib)(brahma)	4008	7.6	333	0	2	30	17/13=1.30
12 PM(jio network)(lohit)	3758	97.5	341	0	4	31	18/13=1.38
3 PM(airtel network)(manas)	4852	47	338	0	2	34	19/15=1.26

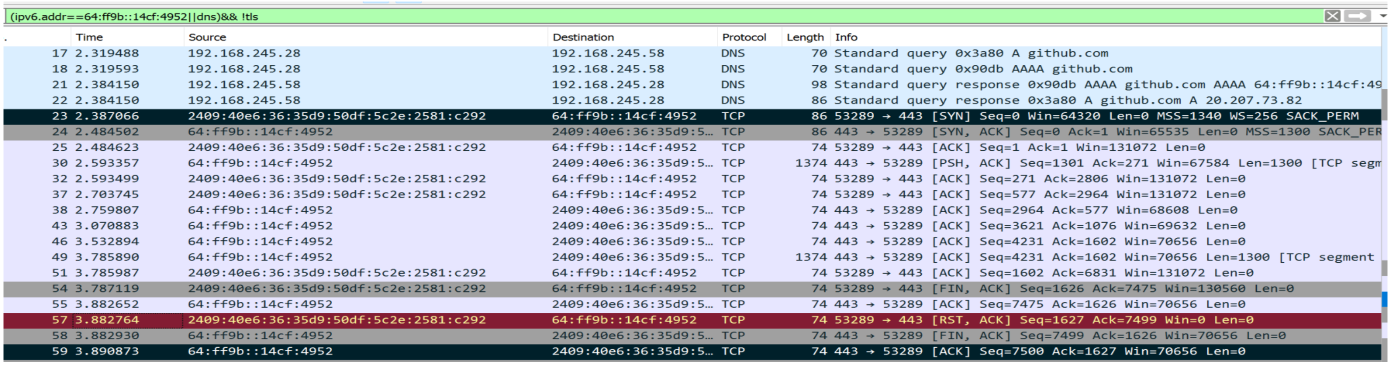
The screenshots of trace files: Link of traces(google drive)

1) 11 AM(lib)(brahma)



```
 938 3.042038  10.150.32.234  172.17.1.1     DNS   70 Standard query 0x4510 A github.com
 939 3.046708  172.17.1.1     10.150.32.234  DNS  501 Standard query response 0x4510 A github.com A 20.207.73.82 NS ns-421
 940 3.050321  10.150.32.234  20.207.73.82   TCP   66 51398 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
 941 3.057910  20.207.73.82   10.150.32.234  TCP   66 443 → 51398 [SYN, ACK] Seq=0 Ack=1 Win=18352 Len=0 MSS=9176 SACK_PER
 942 3.058017  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
 944 3.077918  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=1 Ack=271 Win=19456 Len=0
1117 3.442379  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [ACK] Seq=271 Ack=2806 Win=65536 Len=0
1123 3.452504  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=2806 Ack=335 Win=19456 Len=0
1124 3.452504  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=2806 Ack=577 Win=20608 Len=0
1145 3.502296  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [ACK] Seq=577 Ack=2964 Win=65536 Len=0
1249 3.758747  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=3618 Ack=1076 Win=21632 Len=0
1317 4.061902  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=4225 Ack=1602 Win=22656 Len=0
1439 4.367179  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [ACK] Seq=1602 Ack=7109 Win=65536 Len=0
1443 4.368628  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [FIN, ACK] Seq=1626 Ack=7473 Win=65280 Len=0
1444 4.370581  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [ACK] Seq=7473 Ack=1626 Win=22656 Len=0
1445 4.370581  20.207.73.82   10.150.32.234  TCP   60 443 → 51398 [FIN, ACK] Seq=7473 Ack=1627 Win=22656 Len=0
1446 4.370710  10.150.32.234  20.207.73.82   TCP   54 51398 → 443 [ACK] Seq=1627 Ack=7474 Win=65280 Len=0
2215 6.492754  10.150.32.234  172.17.1.2     DNS   89 Standard query 0x26a3 A avatars.githubusercontent.com
```

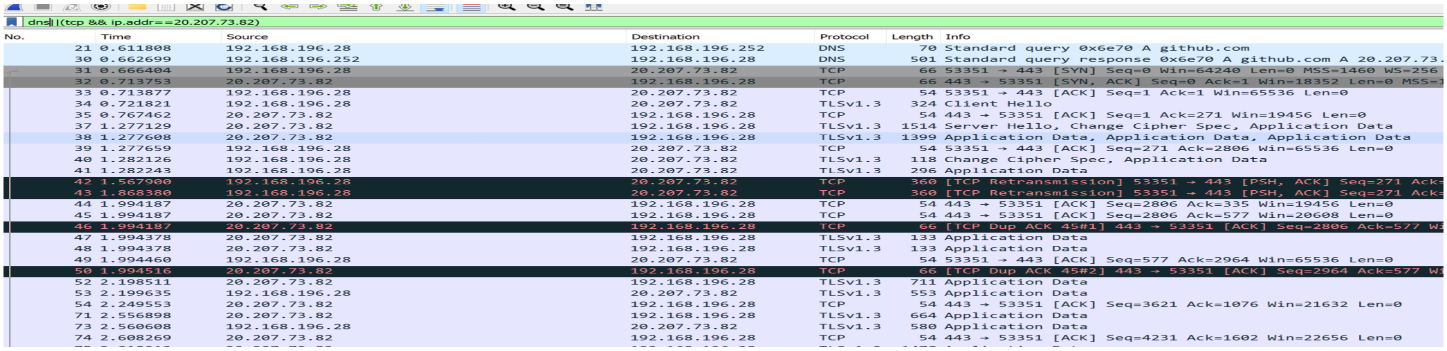
2) 12 PM(jio network)(lohit)



```
(ipv6.addr==64:ff9b::14cf:4952||dns)&& !tls

    Time      Source                                  Destination              Protocol Length Info
 17 2.319488  192.168.245.28                          192.168.245.58           DNS      70 Standard query 0x3a80 A github.com
 18 2.319593  192.168.245.28                          192.168.245.58           DNS      70 Standard query 0x90db AAAA github.com
 21 2.384150  192.168.245.58                          192.168.245.28           DNS      98 Standard query response 0x90db AAAA github.com AAAA 64:ff9b::14cf:49
 22 2.384150  192.168.245.58                          192.168.245.28           DNS      86 Standard query response 0x3a80 A github.com A 20.207.73.82
 23 2.387066  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      86 53289 → 443 [SYN] Seq=0 Win=64320 Len=0 MSS=1340 WS=256 SACK_PERM
 24 2.484502  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      86 443 → 53289 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1300 SACK_PER
 25 2.484623  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
 30 2.593357  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP    1374 443 → 53289 [PSH, ACK] Seq=1301 Ack=271 Win=67584 Len=1300 [TCP segm
 32 2.593499  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [ACK] Seq=271 Ack=2806 Win=131072 Len=0
 37 2.703745  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [ACK] Seq=577 Ack=2964 Win=131072 Len=0
 38 2.759807  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [ACK] Seq=2964 Ack=577 Win=68608 Len=0
 43 3.070883  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [ACK] Seq=3621 Ack=1076 Win=69632 Len=0
 46 3.532894  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [ACK] Seq=4231 Ack=1602 Win=70656 Len=0
 49 3.785890  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP    1374 443 → 53289 [ACK] Seq=4231 Ack=1602 Win=70656 Len=1300 [TCP segment
 51 3.785987  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [ACK] Seq=1602 Ack=6831 Win=131072 Len=0
 54 3.787119  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [FIN, ACK] Seq=1626 Ack=7475 Win=130560 Len=0
 55 3.882652  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [ACK] Seq=7475 Ack=1626 Win=70656 Len=0
 57 3.882764  2409:40e6:36:35d9:50df:5c2e:2581:c292   64:ff9b::14cf:4952       TCP      74 53289 → 443 [RST, ACK] Seq=1627 Ack=7499 Win=0 Len=0
 58 3.882930  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [FIN, ACK] Seq=7499 Ack=1626 Win=70656 Len=0
 59 3.890873  64:ff9b::14cf:4952                      2409:40e6:36:35d9:5…     TCP      74 443 → 53289 [ACK] Seq=7500 Ack=1627 Win=70656 Len=0
```

3) 3 PM(airtel network)(manas)



```
dns||(tcp && ip.addr==20.207.73.82)

No. Time      Source             Destination        Protocol Length Info
 21 0.611808  192.168.196.28     192.168.196.252    DNS       70 Standard query 0x6e70 A github.com
 30 0.662699  192.168.196.252    192.168.196.28     DNS      501 Standard query response 0x6e70 A github.com A 20.207.73.
 31 0.666404  192.168.196.28     20.207.73.82       TCP       66 53351 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256
 32 0.713753  20.207.73.82       192.168.196.28     TCP       66 443 → 53351 [SYN, ACK] Seq=0 Ack=1 Win=18352 Len=0 MSS=1
 33 0.713877  192.168.196.28     20.207.73.82       TCP       54 53351 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
 34 0.721821  192.168.196.28     20.207.73.82       TLSv1.3  324 Client Hello
 35 0.767462  20.207.73.82       192.168.196.28     TCP       54 443 → 53351 [ACK] Seq=1 Ack=271 Win=19456 Len=0
 37 1.277129  20.207.73.82       192.168.196.28     TLSv1.3 1514 Server Hello, Change Cipher Spec, Application Data
 38 1.277608  20.207.73.82       192.168.196.28     TLSv1.3 1399 Application Data, Application Data, Application Data
 39 1.277659  192.168.196.28     20.207.73.82       TCP       54 53351 → 443 [ACK] Seq=271 Ack=2806 Win=65536 Len=0
 40 1.282126  192.168.196.28     20.207.73.82       TLSv1.3  118 Change Cipher Spec, Application Data
 41 1.282243  192.168.196.28     20.207.73.82       TLSv1.3  296 Application Data
 42 1.567900  192.168.196.28     20.207.73.82       TCP      360 [TCP Retransmission] 53351 → 443 [PSH, ACK] Seq=271 Ack=
 43 1.868380  192.168.196.28     20.207.73.82       TCP      360 [TCP Retransmission] 53351 → 443 [PSH, ACK] Seq=271 Ack=
 44 1.994187  20.207.73.82       192.168.196.28     TCP       54 443 → 53351 [ACK] Seq=2806 Ack=335 Win=19456 Len=0
 45 1.994187  20.207.73.82       192.168.196.28     TCP       54 443 → 53351 [ACK] Seq=2806 Ack=577 Win=20608 Len=0
 46 1.994187  20.207.73.82       192.168.196.28     TCP       66 [TCP Dup ACK 45#1] 443 → 53351 [ACK] Seq=2806 Ack=577 Wi
 47 1.994378  20.207.73.82       192.168.196.28     TLSv1.3  133 Application Data
 48 1.994378  20.207.73.82       192.168.196.28     TLSv1.3  133 Application Data
 49 1.994460  192.168.196.28     20.207.73.82       TCP       54 53351 → 443 [ACK] Seq=577 Ack=2964 Win=65536 Len=0
 50 1.994516  20.207.73.82       192.168.196.28     TCP       66 [TCP Dup ACK 45#2] 443 → 53351 [ACK] Seq=2964 Ack=577 Wi
 52 2.198511  20.207.73.82       192.168.196.28     TLSv1.3  711 Application Data
 53 2.199635  192.168.196.28     20.207.73.82       TLSv1.3  553 Application Data
 54 2.249553  20.207.73.82       192.168.196.28     TCP       54 443 → 53351 [ACK] Seq=3621 Ack=1076 Win=21632 Len=0
 71 2.556898  20.207.73.82       192.168.196.28     TLSv1.3  664 Application Data
 73 2.560608  192.168.196.28     20.207.73.82       TLSv1.3  580 Application Data
 74 2.608269  20.207.73.82       192.168.196.28     TCP       54 443 → 53351 [ACK] Seq=4231 Ack=1602 Win=22656 Len=0
```