COMP416: Computer Networks

Project 2 Report

Analysis of Transport Layer Protocols with Wireshark

Meryem Karakaş 69074

Part 1. UDP Experiments

```
C:\Users\5>nslookup www.yale.edu
Server:  ns03.ku.edu.tr
Address:  172.20.18.196

Non-authoritative answer:
Name:    pantheon-systems.map.fastly.net
Addresses:  2a04:4e42:400::645
          2a04:4e42:200::645
          2a04:4e42::645
          2a04:4e42:600::645
          151.101.130.133
          151.101.194.133
          151.101.2.133
          151.101.66.133
Aliases:  www.yale.edu
```

Figure 1: nslookup command for www.yale.edu

1. *How many queries are shown in Wireshark in response to a single nslookup command? If more than one, what could be the reason?*

dns.qry.name == "www.yale.edu"

No.	Time	Source	Destination	Protocol	stream index	Info
4232	14:20:45,531479	172.16.104.126	172.20.18.196	DNS		Standard query 0x0006 A www.yale.edu
4280	14:20:45,666451	172.20.18.196	172.16.104.126	DNS		Standard query response 0x0006 A www.yale.edu CNAME panthe
4281	14:20:45,678487	172.16.104.126	172.20.18.196	DNS		Standard query 0x0007 AAAA www.yale.edu
4282	14:20:45,741377	172.20.18.196	172.16.104.126	DNS		Standard query response 0x0007 AAAA www.yale.edu CNAME par

Figure 2: nslookup queries and responses

There are 2 queries are shown in Wireshark in response to a single nslookup command. The DNS query started with an A record query (No. 4232) followed by an A record response (No. 4280). Then an AAAA record query (No. 4281) was sent and an AAAA record response (No. 4282) was returned. Which means system requests both IPv4 and IPv6. Number of queries depends on the DNS configurations, there can be one or two queries. In case there is an error in the first query, a second query may be sent.

2. *What does the stream index for a UDP packet signify in Wireshark?*

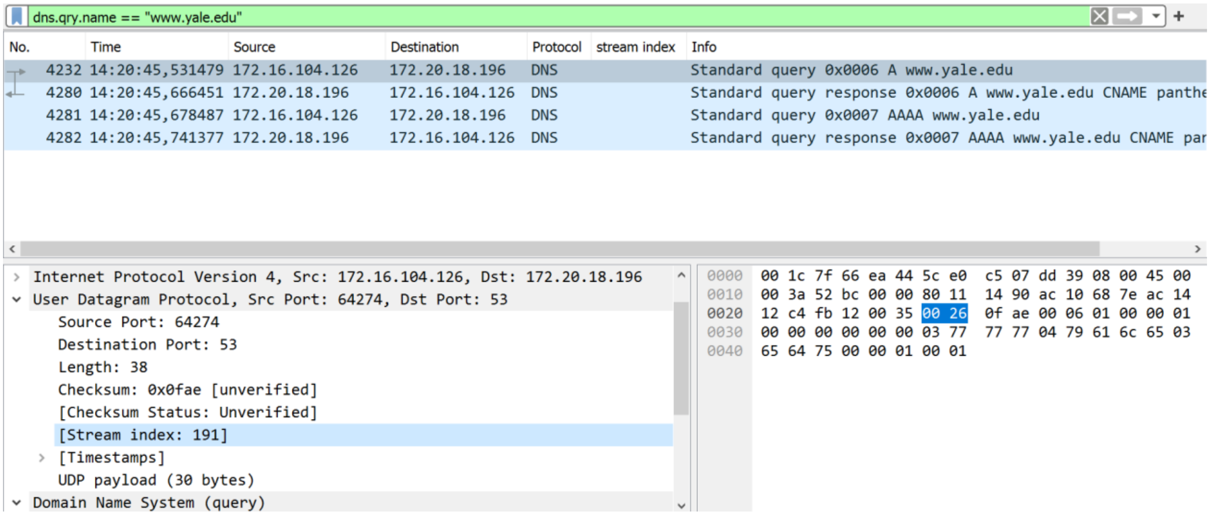


Figure 3: Stream index for UDP

It is unique for each packet in one UDP communication. They are in order, so we can analyze the order of the packets from the stream index. Therefore, the stream index simplifies the debugging and analyzing the communication. Also, it is the same for a query and response of that query in the Wireshark.

3. *Observe the Flowgraph for the UDP messages. Which IP address is the domain name being resolved? Can you apply domain name resolution on that IP address and perform a quick internet search about the DNS server?*

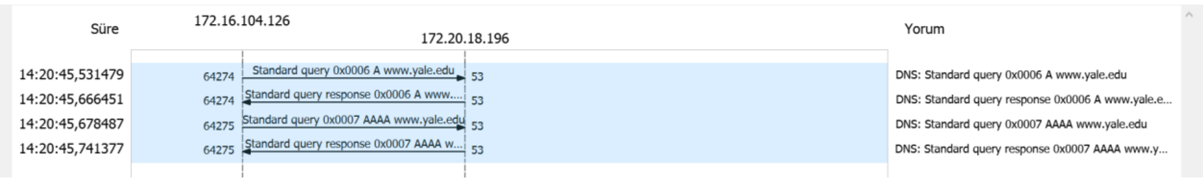


Figure 4: Flowgraph for UDP messages

It queries IP address of the www.yale.edu domain name. We can apply domain name resolution on that IP address over the internet. There are a lot of website which provides domain name to IP address resolution and vice versa. One of them is:

https://whatismyipaddress.com/hostname-ip

4. *Observe the flags under the DNS header. What is the purpose of the Authoritative and Recursion flags?*

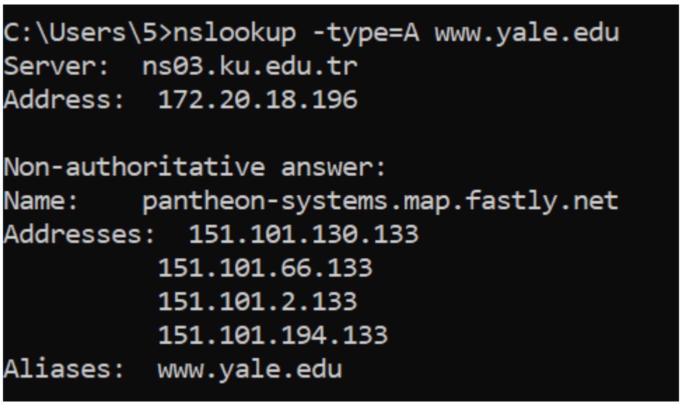
The Authoritative flag indicates that the DNS server is an authoritative source, the server doesn't need to consult any higher authority to serve responses. Conversely, the Recursion flag indicates that the DNS server performs queries recursively. Recursive servers perform queries by consulting higher authorities.

5. *What are the types of DNS Records (name them in the report, but you may be asked about their significance during the demo)? How can you specify the type of DNS Record when using the 'nslookup' command? Share the results using the nslookup with any '3' DNS Record Types.*

DNS record types are records that provide important information about a hostname or domain. The following are the major DNS record types:

- A record: shows the IP address for a specific hostname or domain name
- AAAA record: points to IPv6 addresses for a domain
- CNAME record: a DNS record that points a alias domain name to cononical domain name
- Nameserver (NS) record: specifies the authoritative DNS server for a domain
- Mail exchange (MX) record: shows where emails for a domain should be routed to

We can specify the type of DNS record by using "-type" flag when using nslookup.



```
C:\Users\5>nslookup -type=A www.yale.edu
Server:  ns03.ku.edu.tr
Address:  172.20.18.196

Non-authoritative answer:
Name:    pantheon-systems.map.fastly.net
Addresses:  151.101.130.133
          151.101.66.133
          151.101.2.133
          151.101.194.133
Aliases:  www.yale.edu
```

Figure 5: nslookup for type A

```
C:\Users\5>nslookup -type=NS www.yale.edu
Server:  ns03.ku.edu.tr
Address:  172.20.18.196

Non-authoritative answer:
www.yale.edu    canonical name = pantheon-systems.map.fastly.net

fastly.net
        primary name server = ns1.fastly.net
        responsible mail addr = hostmaster.fastly.com
        serial  = 2017052201
        refresh = 3600 (1 hour)
        retry   = 600 (10 mins)
        expire  = 604800 (7 days)
        default TTL = 30 (30 secs)
```

Figure 6: nslookup for type NS

```
C:\Users\5>nslookup -type=CNAME www.yale.edu
Server:  ns03.ku.edu.tr
Address:  172.20.18.196

Non-authoritative answer:
www.yale.edu    canonical name = pantheon-systems.map.fastly.net

yale.edu        nameserver = ns0116.secondary.cloudflare.com
yale.edu        nameserver = ns0146.secondary.cloudflare.com
yale.edu        nameserver = pks1302-102.net.yale.edu
yale.edu        nameserver = pks1302-103.net.yale.edu
ns0116.secondary.cloudflare.com internet address = 162.159.32.117
ns0116.secondary.cloudflare.com AAAA IPv6 address = 2606:4700:51::a29f:2075
ns0146.secondary.cloudflare.com internet address = 162.159.33.21
pks1302-102.net.yale.edu        internet address = 128.36.72.27
pks1302-103.net.yale.edu        internet address = 128.36.72.29
```

Figure 7: nslookup for type CNAME

Part 2. TCP Experiments

http

No.	Time	Source	Destination	Protocol	stream index	Info
875	15:34:33,463635	172.16.104.126	128.119.245.12	HTTP	6	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plai
914	15:34:33,602717	128.119.245.12	172.16.104.126	HTTP	6	HTTP/1.1 200 OK (text/html)

Figure 8: HTTP POST request and response

Three way TCP handshake at the beginning:

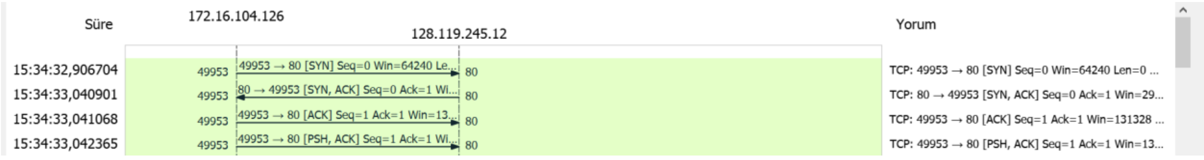


Figure 9: TCP handshake

Closing connection after getting response:

Figure 10: TCP termination

6. *Use the Flow graph options under Statistics in Wireshark and identify the major sequences (handshake, termination, data exchange, etc.) concerning packets involved in exchanging information. Use the following format*

Start-End time	Source Socket	Destination Socket	Sequence Number	Number of Packets in the stage	Stream ID

3-Way handshake:

Start Time: 15:34:32

End Time: 15:34:33

Source Socket: (IP: 172.16.104.126) (Port: 49953)

Destination Socket: (IP: 128.119.245.12) (Port: 80)

Sequence Number: 0,1

Number of packets in the stage: 3

Stream ID: 6

Data exchange: transfer the file to a Web server using the HTTP POST method

Start Time: 15:34:33

End Time: 15:34:33

Source Socket: (IP: 172.16.104.126) (Port: 49953)

Destination Socket: (IP: 128.119.245.12) (Port: 80)

Sequence Number: between 1-153050 for file sending side, between 1-777 for remote server side

Number of packets in the stage:

Stream ID: 6

Termination: closing the connection (I closed the browser after getting http response and sending ACK)

Start Time: 15:34:37

End Time: 15:34:37

Source Socket: (IP: 172.16.104.126) (Port: 49953)

Destination Socket: (IP: 128.119.245.12) (Port: 80)

Sequence Number: 153050,153051 for file sending side, 778 for remote server side

Number of packets in the stage: 3

Stream ID: 6

7. *What does the stream index in the TCP header signify? Are the packets being transmitted during the experiment all belonging to the same stream index? What does the same or different stream index mean in the context of this experiment?*

The stream index specifies the specific TCP stream. A stream index is created for a connection between two systems. For example, if a TCP stream has a stream index, another stream must have another stream index. The packets that are being transmitted during the experiment have the same stream index because they belong to the same TCP stream.

8. *Print the RTT graph for the entire communication. What is the average RTT value for the entire communication sequence?*

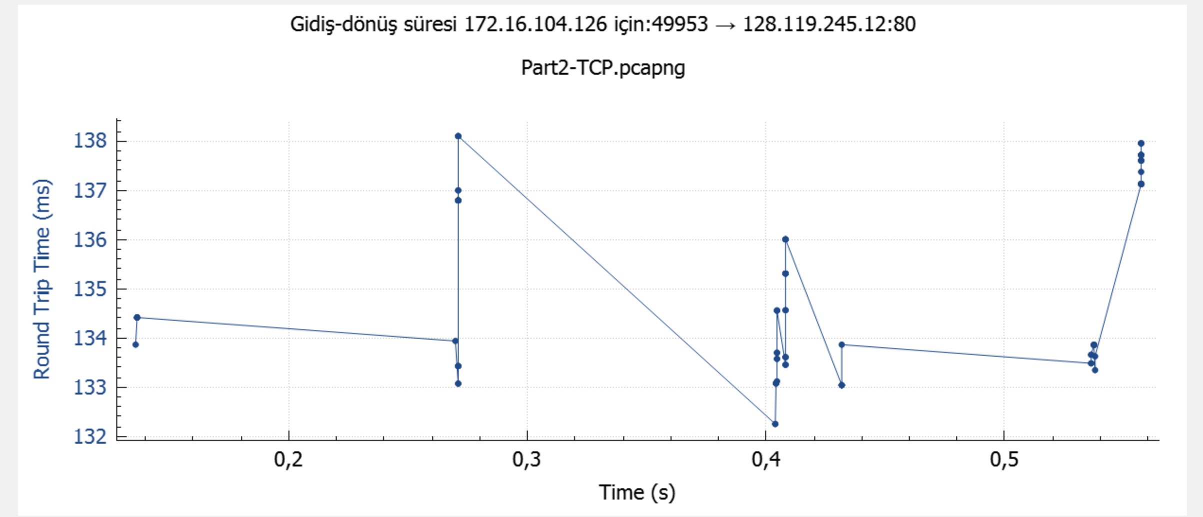


Figure 11: RTT graph

9. *Explain and provide the values for the following fields in the TCP packets:*

(a) TCP Segment Length :
Is the total lenght of header lenght and data length of the segment.
It is the 1490 bytes for segments which consist of chunks of alice.txt file.
It is 0 for ACK messages send from remote web server when it receive data from our machine.
It is 0 for hanshake messages and termination mesages.
So, it can vary depending on the payload. However, as I understood from Wireshark, it is 1460 byte at most.

(b) Sequence Number and relative Sequence Number :
The sequence number of the segment indicates byte-stream number of the first byte int the TCP segment. The relative sequence number indicates the sequence number relative to the initial sequence number in the connection.

(c) Acknowledgement Number and relative Acknowledgement number :
The acknowledgement number of sender indicates that all bytes up to that number have been received from the receiver. The relative acknowledgement number is relative to the initial sequence number of the connection.

(d) Nonce Flag :
The Nonce flag shows whether the segment contains a Nonce. It is used to protect communications.

(e) Congestion Window Reduce (CWR) Flag:

Indicates whether the sender has reduced the congestion window size due to network congestion. Its value is “Not Set” for my TCP connection.

Part 3. A. SSL Implementation and Experiments

10. *What cipher suite set does the client send? Which cipher does the server choose for the remainder of this communication?*

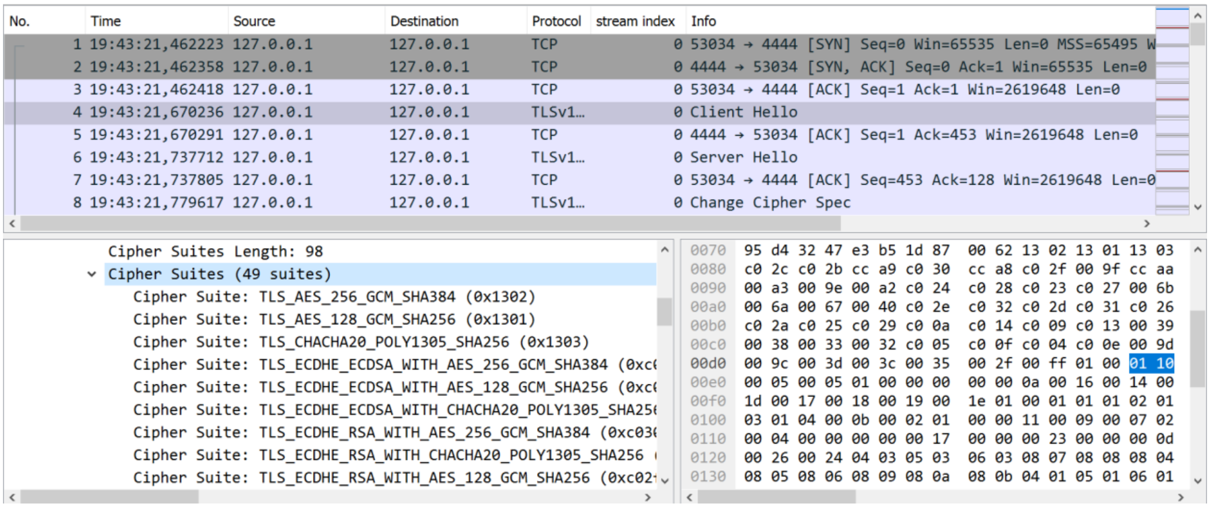


Figure 12: Cipher suites that the client supports

We can find the supported cipher suite set in the Client Hello message. As shown in the figure, the client supports 49 cipher suites. Also, we can find the cipher süite that the server chose in the Server Hello message. As you can in the figure below, it is **TLS_AES_256_GCM_SHA384 (0x1302)**

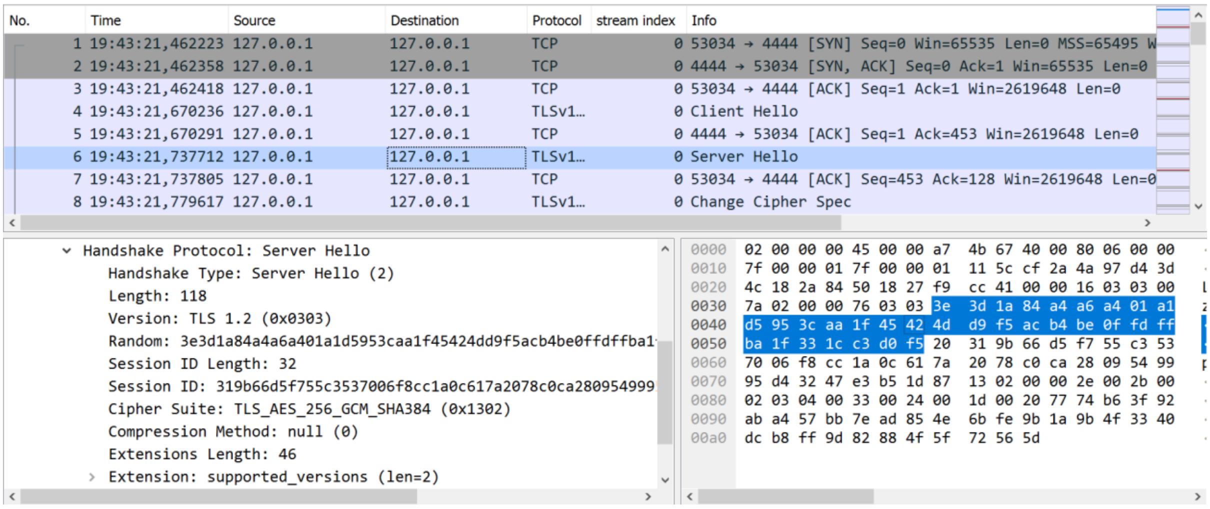


Figure 13: The cipher suite that the server chose

11. *Observe the packet showing the certificate transfer. Check under the SSL header. How many certificates does the server send during the Handshake, and why?*

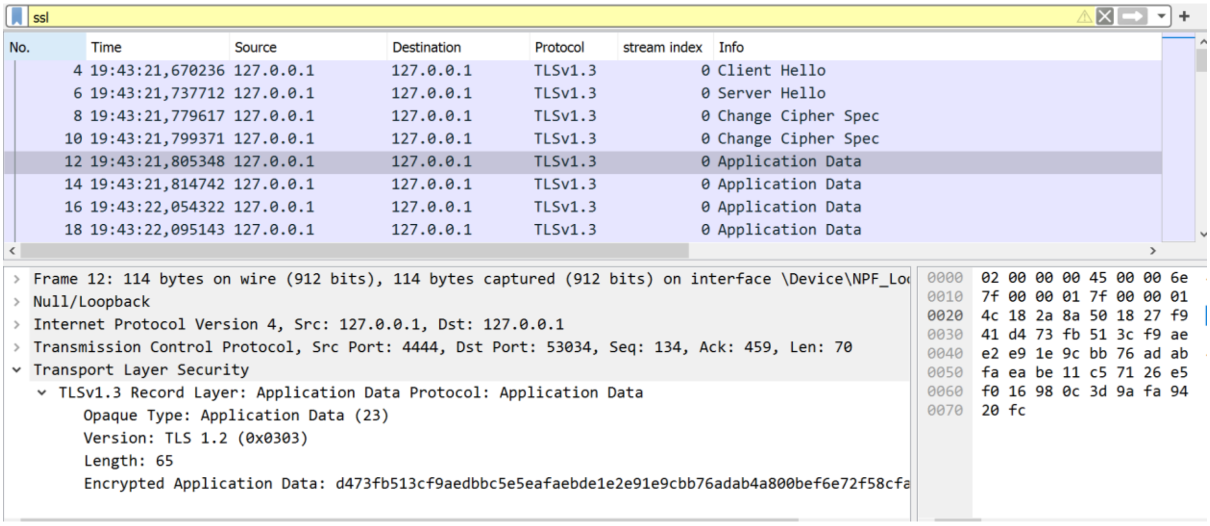


Figure 14: Encrypted Certificate

Certificate is encrypted under Application Data Message.

12. *Find the message from the Server Hello group containing the session ID. What is the session ID in the corresponding message? What is the purpose of specifying a session ID?*

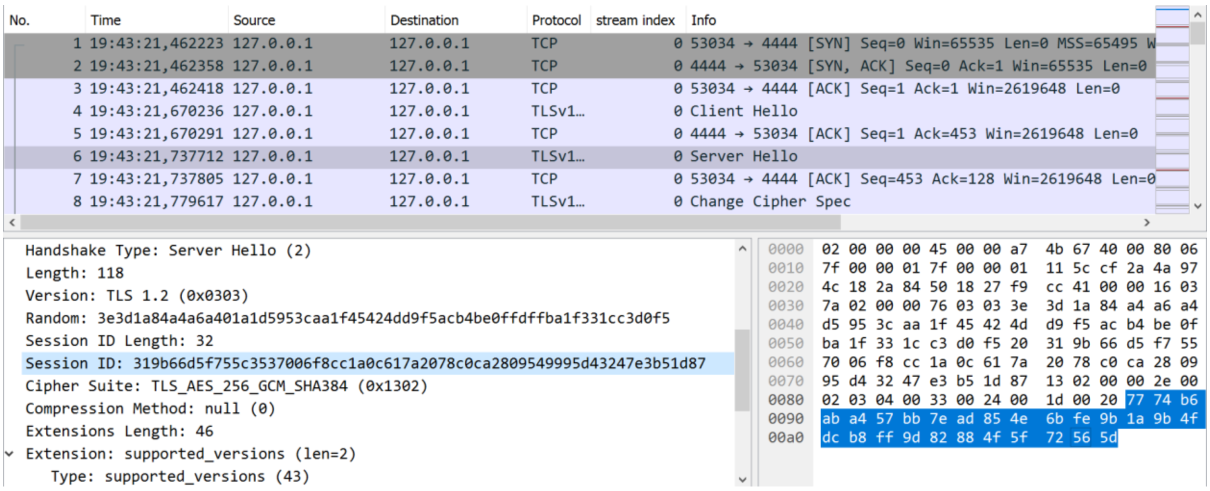


Figure 15: Session ID in the Server Hello message

As you can see in the figure above, the session ID is:
319b66d5f755c3537006f8cc1a0c617a2078c0ca2809549995d43247e3b51d87

Session ID is a unique indetifier for specific session between client and server.It helps server to differentiate between different sessions. Thanks to session ID, the server is able to remembers the negotiated cipher suite and keys from previous handshake and is able to reuse them.

Part 3. B. SSL vs. TCP: Delay Measurements

First, I want to explain my code implementation in this part:

SSL part: When you run the client, it asks you to write 1 or 2 to choose the connection type. If you want a persistent connection, you should write 1, otherwise, you should write 2. If you choose a persistent one, the server generates a random character and sends it to the client with a persistent connection 16 times. Otherwise, the server generates and sends each character one by one with a non-persistent connection. Also, to calculate the time delay, I used two of the java.time types: the Instant and the Duration classes. I started the timer when the client requests a string, and stop it when the sender sends the entire string.

TCP part: I did the same thing I did in the SSL part, just changed the connection type. I used Multi-threading Server-Client codes as a draft.

13. *Report both delays for 5 different executions and present the measurements as a single graph. Briefly describe the reasons for the results you have obtained.*

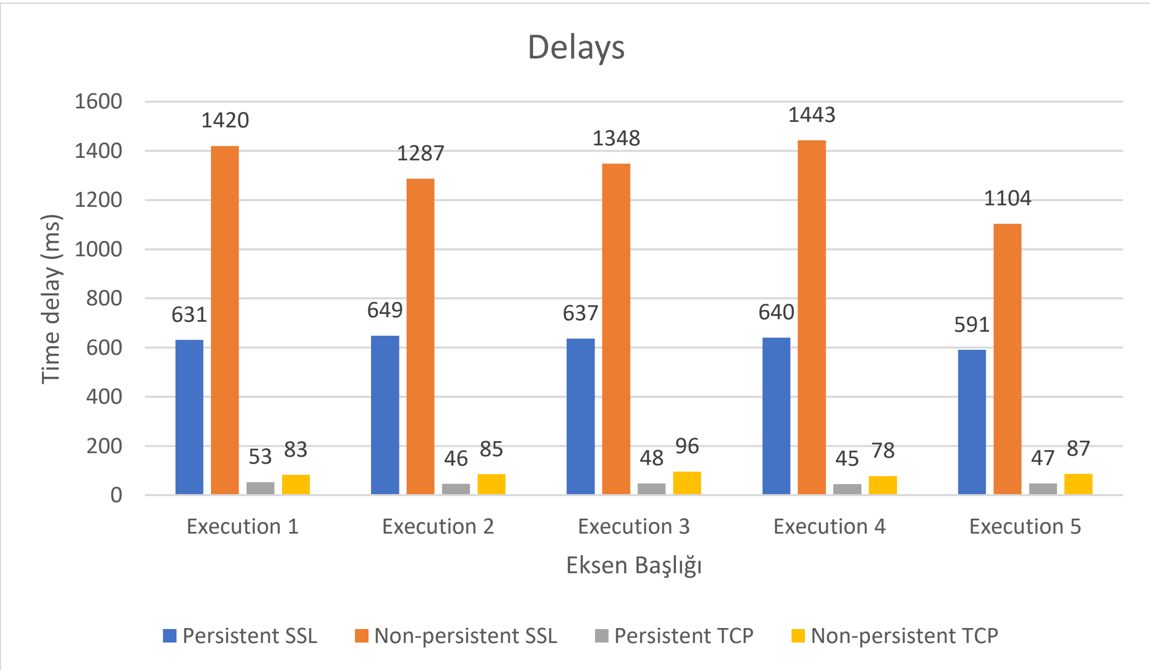
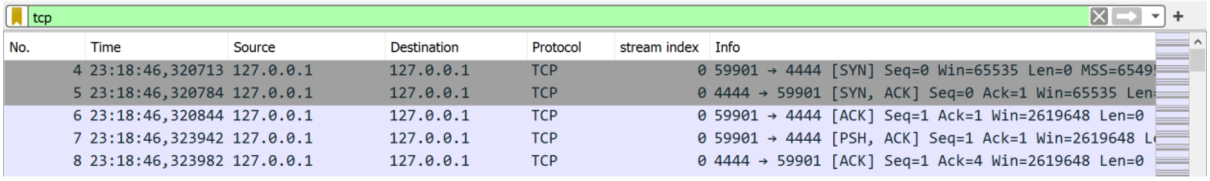


Table 1: Time delay measurements for TCP and SSL

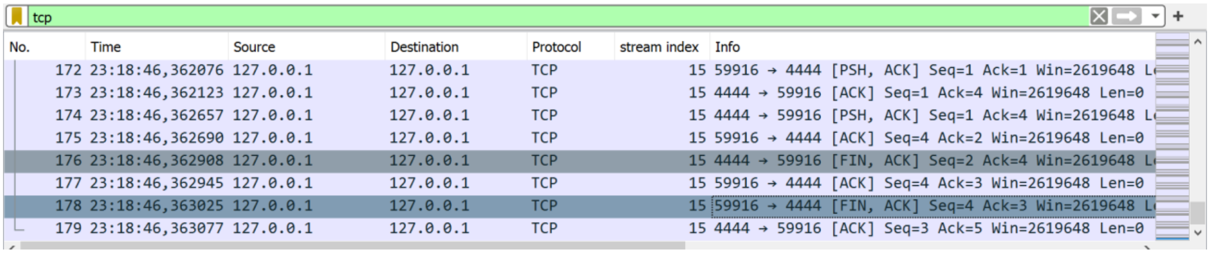
As we expected, persistent connections transmit data in a shorter time than non-persistent connections. Because time is not spent re-establishing the connection in each message, other messages are easily sent over the existing connection. In addition, TCP data transfer is faster than SSL. TCP is faster because it does not require the extra steps of encryption and decryption required by SSL. SSL requires more processing time, which can slow down data transfer.

14. *Find the TCP Handshake and Terminate Messages. Do these messages have the stream index? Why or why not?*



No.	Time	Source	Destination	Protocol	stream index	Info
4	23:18:46,320713	127.0.0.1	127.0.0.1	TCP	0	59901 → 4444 [SYN] Seq=0 Win=65535 Len=0 MSS=6549
5	23:18:46,320784	127.0.0.1	127.0.0.1	TCP	0	4444 → 59901 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len
6	23:18:46,320844	127.0.0.1	127.0.0.1	TCP	0	59901 → 4444 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
7	23:18:46,323942	127.0.0.1	127.0.0.1	TCP	0	59901 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=2619648 L
8	23:18:46,323982	127.0.0.1	127.0.0.1	TCP	0	4444 → 59901 [ACK] Seq=1 Ack=4 Win=2619648 Len=0

Figure 16: Stream index for the TCP hadshake message



No.	Time	Source	Destination	Protocol	stream index	Info
172	23:18:46,362076	127.0.0.1	127.0.0.1	TCP	15	59916 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=2619648 L
173	23:18:46,362123	127.0.0.1	127.0.0.1	TCP	15	4444 → 59916 [ACK] Seq=1 Ack=4 Win=2619648 Len=0
174	23:18:46,362657	127.0.0.1	127.0.0.1	TCP	15	4444 → 59916 [PSH, ACK] Seq=1 Ack=4 Win=2619648 L
175	23:18:46,362690	127.0.0.1	127.0.0.1	TCP	15	59916 → 4444 [ACK] Seq=4 Ack=2 Win=2619648 Len=0
176	23:18:46,362908	127.0.0.1	127.0.0.1	TCP	15	4444 → 59916 [FIN, ACK] Seq=2 Ack=4 Win=2619648 L
177	23:18:46,362945	127.0.0.1	127.0.0.1	TCP	15	59916 → 4444 [ACK] Seq=4 Ack=3 Win=2619648 Len=0
178	23:18:46,363025	127.0.0.1	127.0.0.1	TCP	15	59916 → 4444 [FIN, ACK] Seq=4 Ack=3 Win=2619648 L
179	23:18:46,363077	127.0.0.1	127.0.0.1	TCP	15	4444 → 59916 [ACK] Seq=3 Ack=5 Win=2619648 Len=0

Figure 17: Stream index for the TCP termination message

Yes, TCP Handshake Messages have stream index 0 and Terminate Messages have stream index 15. For example, there is more than one TCP handshake and termination in a non-persistent connection. In such cases, it may be necessary to assign an index to them in order to determine which hanshake/termination message belongs to which stream.