

CS433 Assignment 1

Shrreya Singh

19110136

Part 1:

GitHub Link: https://github.com/ShrreyaSingh/CS433_A1_RFS.git

c) Wireshark dump and analysis indicating data was correctly encrypted.

1. Checking Substitution Encryption:

Command: cd server_files

The actual encrypted command sent to the server and the encrypted response received are shown below (both hex and encrypted strings):

```
[(base) shrreyasinhg@Shrreyas-MacBook-Pro cs433_A1_rfs % python3 c_app.py

Enter any one of the below commands:
CWD
LS
CD dir_path
DWD server_file_path
UPD client_file_path

-----
cd server_files

Enter an integer to select the encryption type:
1 Plain
2 Substitute
3 Transpose

-----
2

Setting up connection ...

Client Request Encrypted String: 2^@^5@^ef@^ugtxgt_hkngu
Hex string = 325e405e3540404065664040407567747867745f686b6e6775

Server Response Encrypted String: 2^@^3^@^Fktgevqta ejcpigf uweeguuhwenna
Hex string = 325e405e335e405e466b7467657671746120656a63706967662075776565677568776e6e61

Connection closed :)
-----
Received: 'Directory changed successfully'
-----
STATUS: OK
```

These results can be verified from the wireshark dump (Q1_A1_cd.pcapng). The snapshots of the request and response payloads (from WireShark dump) match the encrypted string and its hex code.

Client Request Payload:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	44572 → 2500 [SYN] Seq=0 Win=65495 Len=100
2	0.000044042	127.0.0.1	127.0.0.1	TCP	76	2500 → 44572 [SYN, ACK] Seq=0 Ack=1 Win=655
3	0.000087917	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000171375	127.0.0.1	127.0.0.1	TCP	93	44572 → 2500 [PSH, ACK] Seq=1 Ack=1 Win=655
5	0.000176709	127.0.0.1	127.0.0.1	TCP	68	2500 → 44572 [ACK] Seq=1 Ack=26 Win=65
6	0.000734375	127.0.0.1	127.0.0.1	TCP	106	2500 → 44572 [PSH, ACK] Seq=1 Ack=26 Win=65
7	0.000739125	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=26 Ack=39 Win=6
8	0.000851542	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [FIN, ACK] Seq=26 Ack=39 Win=6
9	0.000909167	127.0.0.1	127.0.0.1	TCP	68	2500 → 44572 [FIN, ACK] Seq=39 Ack=27
10	0.000911117	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=39 Ack=27 Win=6

[Window size scaling factor: 128]
Checksum: 0xfe41 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
[iRTT: 0.000087917 seconds]
[Bytes in flight: 25]
[Bytes sent since last PSH flag: 25]
TCP payload (25 bytes)

```

0000 00 00 03 04 00 06 00 00 00 00 00 00 00 00 00 08 00
0010 45 00 00 4d 19 57 40 00 40 06 23 52 7f 00 00 01 E - M W@ - @ - #R - ...
0020 7f 00 00 01 ae 1c 09 c4 44 dc 34 bf e9 84 49 7e ..... D - 4 - I -
0030 80 18 02 00 fe 41 00 00 01 01 08 0a 4d 0b 57 55 ..... A - ..... M - WU
0040 4d 0b 57 55 32 5e 40 5e 35 40 40 40 65 66 40 40 M - WU 2^@^ 5@^@ef@^
0050 40 75 67 74 78 67 74 5f 68 6b 6e 67 75 @ugtxgt_ hkngu

```

The TCP payload of this packet (tcp.payload), 25 bytes

Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

Server Response Payload:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	44572 → 2500 [SYN] Seq=0 Win=65495 Len=100
2	0.000044042	127.0.0.1	127.0.0.1	TCP	76	2500 → 44572 [SYN, ACK] Seq=0 Ack=1 Win=655
3	0.000087917	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000171375	127.0.0.1	127.0.0.1	TCP	93	44572 → 2500 [PSH, ACK] Seq=1 Ack=1 Win=655
5	0.000176709	127.0.0.1	127.0.0.1	TCP	68	2500 → 44572 [ACK] Seq=1 Ack=26 Win=65
6	0.000734375	127.0.0.1	127.0.0.1	TCP	106	2500 → 44572 [PSH, ACK] Seq=1 Ack=26 Win=65
7	0.000739125	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=26 Ack=39 Win=6
8	0.000851542	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [FIN, ACK] Seq=26 Ack=39 Win=6
9	0.000909167	127.0.0.1	127.0.0.1	TCP	68	2500 → 44572 [FIN, ACK] Seq=39 Ack=27
10	0.000911117	127.0.0.1	127.0.0.1	TCP	68	44572 → 2500 [ACK] Seq=39 Ack=27 Win=6

[Window size scaling factor: 128]
Checksum: 0xfe4e [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
[iRTT: 0.000087917 seconds]
[Bytes in flight: 38]
[Bytes sent since last PSH flag: 38]
TCP payload (38 bytes)

```

0000 00 00 03 04 00 06 00 00 00 00 00 00 00 00 00 08 00
0010 45 00 00 5a 21 29 40 00 40 06 1b 73 7f 00 00 01 E - Z! )@ - @ - S - ...
0020 7f 00 00 01 09 c4 ae 1c e9 84 49 7e 44 dc 34 d8 ..... I - D - 4 - 
0030 80 18 02 00 fe 4e 00 00 01 01 08 0a 4d 0b 57 56 ..... N - ..... M - WV
0040 4d 0b 57 55 32 5e 40 5e 33 5e 40 5e 46 6b 74 67 M - WU 2^@^ 3@^@Fktg
0050 65 76 71 74 61 20 65 6a 63 70 69 67 66 20 75 77 evqta ej cpigf uw
0060 65 65 67 75 75 68 77 6e 6e 61 eeguuuhw na

```

The TCP payload of this packet (tcp.payload), 38 bytes

Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

2. Checking Transpose Encryption:

Command: dwd server_files/download1.txt

The actual encrypted command sent to the server and the encrypted response received are shown below (both hex and encrypted strings):

```
(base) shrreyasingh@Shrreyas-MacBook-Pro cs433_A1_rfs % python3 c_app.py
Enter any one of the below commands:
CWD
LS
CD dir_path
DWD server_file_path
UPD client_file_path
-----
dwd server_files/download1.txt
Enter an integer to select the encryption type:
1 Plain
2 Substitute
3 Transpose
-----
3
Setting up connection ...
Client Request Encrypted String: 3^@^txt.1daolnwod/selif_revres@@@dwd@@@4
Hex string = 335e405e7478742e3164616f6c6e776f642f73656c69665f726576726573404064776440404034
Server Response Encrypted String: 3^@^noitpyrcnE gnikcehC
): sijome emos htiw
senil elpitium evah I dna
revres eht morf dedaoInwod elif a ma I@@@txt.1daolnwod^@^
Hex string = 335e405e666f6974707972636e45286766696b636568430a293a2073696a6f6d6520656d6f7320687469770a73656e696c20656c7069746c756d2065766168204920646e610a2072657672657320656874206d6f726620646564
616fc6e6776f6420656c69662061206d6120494040407478742e3164616f6c6e776f645e405e31
Connection closed :)
Received: 'I am a file downloaded from the server \nand I have multiple lines\nwith some emojis :)\nChecking Encryption'
STATUS: OK
```

These results can be verified from the wireshark dump (Q1_A1_dwd.pcapng). The snapshots of the request and response payloads (from WireShark dump) match the encrypted string and its hex code.

Client Request Payload:

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000034583	127.0.0.1	127.0.0.1	TCP	76	2500 → 44578 [SYN, ACK] Seq=0 Ack=1 Win=655
3	0.000059041	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000109708	127.0.0.1	127.0.0.1	TCP	108	44578 → 2500 [PSH, ACK] Seq=1 Ack=1 Win=655
5	0.000115833	127.0.0.1	127.0.0.1	TCP	68	2500 → 44578 [ACK] Seq=1 Ack=41 Win=655
6	0.001416458	127.0.0.1	127.0.0.1	TCP	197	2500 → 44578 [PSH, ACK] Seq=1 Ack=41 Win=655
7	0.001455000	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=41 Ack=130 Win=655
8	0.001584625	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [FIN, ACK] Seq=41 Ack=130 Win=655
9	0.002050583	127.0.0.1	127.0.0.1	TCP	68	2500 → 44578 [FIN, ACK] Seq=130 Ack=42 Win=655
10	0.002059083	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=42 Ack=131 Win=655

Checksum: 0xfe50 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
[iRTT: 0.000059041 seconds]
[Bytes in flight: 40]
[Bytes sent since last PSH flag: 40]

TCP payload (40 bytes)

Data (40 bytes)

```

0000 00 00 03 04 00 06 00 00 00 00 00 00 00 00 00 08 00 ..... .
0010 45 00 00 5c 7d a3 40 00 40 06 be f6 7f 00 00 01 E.. \} .@. @.....
0020 7f 00 00 01 ae 22 09 c4 5b e8 81 ef 01 62 32 3d ..... " .. [ ... b2= ...
0030 80 18 02 00 fe 50 00 00 01 01 08 0a 4d 1d 3f c7 ..... P. .... M. ? - .
0040 4d 1d 3f c7 33 5e 40 5e 74 78 74 2e 31 64 61 6f M. ? 3@^ txt.1dao
0050 6c 6e 77 6f 64 2f 73 65 6c 69 66 5f 72 65 76 72 lnwod/se lif_revr
0060 65 73 40 40 40 64 77 64 40 40 40 34 es@@dwd 00@4

```

The TCP payload of this packet (tcp.payload), 40 bytes

Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

Server Response Payload:

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000034583	127.0.0.1	127.0.0.1	TCP	76	2500 → 44578 [SYN, ACK] Seq=0 Ack=1 Win=655
3	0.000059041	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000109708	127.0.0.1	127.0.0.1	TCP	108	44578 → 2500 [PSH, ACK] Seq=1 Ack=1 Win=655
5	0.000115833	127.0.0.1	127.0.0.1	TCP	68	2500 → 44578 [ACK] Seq=1 Ack=41 Win=655
6	0.001416458	127.0.0.1	127.0.0.1	TCP	197	2500 → 44578 [PSH, ACK] Seq=1 Ack=41 Win=655
7	0.001455000	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=41 Ack=130 Win=655
8	0.001584625	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [FIN, ACK] Seq=41 Ack=130 Win=655
9	0.002050583	127.0.0.1	127.0.0.1	TCP	68	2500 → 44578 [FIN, ACK] Seq=130 Ack=42 Win=655
10	0.002059083	127.0.0.1	127.0.0.1	TCP	68	44578 → 2500 [ACK] Seq=42 Ack=131 Win=655

Checksum: 0xfea9 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
[iRTT: 0.000059041 seconds]
[Bytes in flight: 129]
[Bytes sent since last PSH flag: 129]

TCP payload (129 bytes)

Data (data), 129 bytes

```

0020 7f 00 00 01 09 c4 ae 22 01 62 32 3d 5b e8 82 17 ..... " b2=[...
0030 80 18 02 00 fe a9 00 00 01 01 08 0a 4d 1d 3f c8 ..... . . . M. ? - .
0040 4d 1d 3f c7 33 5e 40 5e 6e 6f 69 74 70 79 72 63 M. ? 3@^ noitpyrc
0050 6e 45 20 67 6e 69 6b 63 65 68 43 0a 29 3a 20 73 nE gnikc ehC. : s
0060 69 6a 6f 6d 65 20 65 6d 6f 73 20 68 74 69 77 0a ijome em os htiw.
0070 73 65 6e 69 6c 20 65 6c 70 69 74 6c 75 6d 20 65 senil el pitlum e
0080 76 61 68 20 49 20 64 6e 61 0a 20 72 65 76 72 65 vah I dn a. revre
0090 73 20 65 68 74 20 6d 6f 72 66 20 64 65 64 61 6f s eht mo rf dedao
00a0 6c 6e 77 6f 64 20 65 6c 69 66 20 61 20 6d 61 20 lnwod/el if a ma
00b0 49 40 40 40 74 78 74 2e 31 64 61 6f 6c 6e 77 6f I@@@txt. 1daolnwo
00c0 64 5e 40 5e 31 d@^@1

```

Data (data), 129 bytes

Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

3. Checking Plain Text:

Command: upd client_files/upload1.txt

The actual encrypted command sent to the server and the encrypted response received are shown below (both hex and encrypted strings):

```
[(base) shrreyasingh@Shrreyas-MacBook-Pro cs433_A1_rfs % python3 c_app.py

Enter any one of the below commands:
CWD
LS
CD dir_path
DWD server_file_path
UPD client_file_path

-----
upd client_files/upload1.txt

Enter an integer to select the encryption type:
1 Plain
2 Substitute
3 Transpose

-----
1

Setting up connection ...

Client Request Encrypted String: 1^@^5@0@upd@0@upload1.txt@@@Hello World
Checking Encryption

Hex string = 315e405e3540404075706440404075706c6f6164312e74787440404048656c6c6f20576f726c640a436865636b696e6720456e6372797074696f6e

Server Response Encrypted String: 1^@^1^@^server_files/upload1.txt

Hex string = 315e405e315e405e7365727665725f66696c65732f75706c6f6164312e747874

Connection closed :)
-----
Received: 'server_files/upload1.txt'
-----
STATUS: OK
```

These results can be verified from the wireshark dump (Q1_A1_upd.pcapng). The snapshots of the request and response payloads (from WireShark dump) match the encrypted string and its hex code.

Client Request Payload:

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000065333	127.0.0.1	127.0.0.1	TCP	76	2500 → 44580 [SYN, ACK] Seq=0 Ack=1 Wi
3	0.000091333	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000160292	127.0.0.1	127.0.0.1	TCP	127	44580 → 2500 [PSH, ACK] Seq=1 Ack=1 Wi
5	0.000164750	127.0.0.1	127.0.0.1	TCP	68	2500 → 44580 [ACK] Seq=1 Ack=60 Win=65
6	0.002532500	127.0.0.1	127.0.0.1	TCP	100	2500 → 44580 [PSH, ACK] Seq=1 Ack=60 W
7	0.002540250	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=60 Ack=33 Win=6
8	0.002675000	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [FIN, ACK] Seq=60 Ack=33
9	0.002823417	127.0.0.1	127.0.0.1	TCP	68	2500 → 44580 [FIN, ACK] Seq=33 Ack=61
10	0.002830833	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=61 Ack=34 Win=6

Checksum: 0xfe63 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▶ [Timestamps]
- [SEQ/ACK analysis]
 [iRTT: 0.000091333 seconds]
 [Bytes in flight: 59]
 [Bytes sent since last PSH flag: 59]
TCP payload (59 bytes)

Data (59 bytes)		
0000	00 00 03 04 00 06 00 00 00 00 00 00 00 00 08 00
0010	45 00 00 6f cc d3 40 00 40 06 6f b3 7f 00 00 01	E..o..@. @o.....
0020	7f 00 00 01 ae 24 09 c4 57 31 93 d5 40 4b b0 bf\$..W1..@K..
0030	80 18 02 00 fe 63 00 00 01 01 08 0a 4d 22 43 eac.....M"C.
0040	4d 22 43 e9 31 5e 40 5e 35 40 40 40 75 70 64 40	M"C1^@^ 50@upd@
0050	40 40 75 70 6c 6f 61 64 31 2e 74 78 74 40 40 40	@@upload 1.txt@@
0060	48 65 6c 6c 6f 20 57 6f 72 6c 64 0a 43 68 65 63	Hello Wo rld.Che
0070	6b 69 6e 67 20 45 6e 63 72 79 70 74 69 6f 6e	king Enc ryption

Data (data), 59 bytes Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) Profile: Default

Server Response Payload:

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000065333	127.0.0.1	127.0.0.1	TCP	76	2500 → 44580 [SYN, ACK] Seq=0 Ack=1 Wi
3	0.000091333	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=1 Ack=1 Win=655
4	0.000160292	127.0.0.1	127.0.0.1	TCP	127	44580 → 2500 [PSH, ACK] Seq=1 Ack=1 Wi
5	0.000164750	127.0.0.1	127.0.0.1	TCP	68	2500 → 44580 [ACK] Seq=1 Ack=60 Win=65
6	0.002532500	127.0.0.1	127.0.0.1	TCP	100	2500 → 44580 [PSH, ACK] Seq=1 Ack=60 W
7	0.002540250	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=60 Ack=33 Win=6
8	0.002675000	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [FIN, ACK] Seq=60 Ack=33
9	0.002823417	127.0.0.1	127.0.0.1	TCP	68	2500 → 44580 [FIN, ACK] Seq=33 Ack=61
10	0.002830833	127.0.0.1	127.0.0.1	TCP	68	44580 → 2500 [ACK] Seq=61 Ack=34 Win=6

Checksum: 0xfe48 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▶ [Timestamps]
- [SEQ/ACK analysis]
 [iRTT: 0.000091333 seconds]
 [Bytes in flight: 32]
 [Bytes sent since last PSH flag: 32]
TCP payload (32 bytes)

Data (32 bytes)		
0000	00 00 03 04 00 06 00 00 00 00 00 00 00 00 08 00
0010	45 00 00 54 b3 af 40 00 40 06 88 f2 7f 00 00 01	E..T..@. @.....
0020	7f 00 00 01 09 c4 ae 24 40 4b b0 bf 57 31 94 10\$ @K..W1..
0030	80 18 02 00 fe 48 00 00 01 01 08 0a 4d 22 43 ecH.....M"C.
0040	4d 22 43 ea 31 5e 40 5e 31 5e 40 5e 73 65 72 76	M"C1^@^ 1@^serv
0050	65 72 5f 66 69 6c 65 73 2f 75 70 6c 6f 61 64 31	er.files /upload1
0060	2e 74 78 74	.txt

The TCP payload of this packet (tcp.payload), 32 bytes Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) Profile: Default

Analysis:

- In all the requests and responses, TCP protocol is used in accordance with our socket definition parameters of clients and servers (SOCK_STREAM).
- In case of encryption (both substitution and transpose) as well as no encryption, the data sent over the network (payload - both hex value and string), as captured by Wireshark, exactly corresponds to the encrypted client request and server response (when explicitly printed out to the terminal before transmission).
- The cross verification is performed on three different commands. Thus, the crypto layer is functioning well in our Remote File System.
- The Wireshark dump for all the three cases are present in the above mentioned repository.

Part 2:

Answer 2

Part 2

Link 2 has min rate of transmission. Therefore, it'll be the bottle neck of our network.

$$\therefore \text{Total time to transmit } n \text{ packets} = \frac{nL}{R_{\text{Link 2}}} + \underbrace{\frac{L}{R_{\text{Link 1}}}}_{\substack{\rightarrow \text{packet size} \\ \text{Link 2}}} + \underbrace{\frac{L}{R_{\text{Link 3}}}}_{\substack{\rightarrow \text{Link 1} \\ \text{Link 3}}}$$

Processing & propagation delay = 0. Sending the last packet.

a) 1 packet, $n = 1$

$$L = 100 kB + 100 B = 100100 B$$

$$t_{\text{delivery}} = \frac{1 \times 100.1 \times 10^3}{100 \times 10^6} + 8 \times 100.1 \times 10^3 \left[\frac{1}{400 \times 10^6} + \frac{1}{200 \times 10^6} \right]$$

$$= 14.014 \text{ ms.}$$

b) 10 packets, $n = 10$

$$L = \frac{100kB + 100B}{10} \Rightarrow L = 10.1 \times 10^3 B$$

$$t_{\text{delivery}} = 10.1 \times 10^3 \times 8 \left[\frac{10}{100 \times 10^6} + \frac{1}{4 \times 10^8} + \frac{1}{2 \times 10^8} \right]$$

$$= 8.686 \text{ ms.}$$

c) 50 packets, $n = 50$.

$$L = \frac{2100kB}{50} + 100B = 2.1 kB$$

$$t_{\text{delivery}} = 2.1 \times 10^3 \times 8 \left[\frac{50}{10^8} + \frac{1}{4 \times 10^8} + \frac{1}{2 \times 10^8} \right]$$

$$= 8.526 \text{ ms}$$

d) 100 packets, $n = 100$

$$L = \frac{100 \text{ kB}}{100} + 100 \text{ B} = 1.1 \text{ kB}$$

$$t_{\text{delivery}} = \frac{1.1 \times 10^3 \times 8}{\frac{100}{10^8} + \frac{1}{4 \times 10^6} + \frac{1}{2 \times 10^8}} \\ = 8.526 \text{ ms}$$

∴ Configuration c gives lowest delivery time (8.526 ms)

3. a) $t_{\text{propagation}} = \frac{\text{len(link)}}{v_{\text{propagation}}} = \frac{M}{s}$

$$= \frac{10 \times 10^3}{2/3 \times 3 \times 10^8} \Rightarrow 5 \times 10^{-5} \text{ s.}$$

= 50 μs . [Propagation Delay]

b) Max bits that R_1 can send before 1st bit reaches R_2 = $\frac{RM}{s}$

$$= \frac{100 \times 10^9}{2 \times 10^8} \times \frac{5}{10 \times 10^3}$$

$$S = \frac{2}{3} C \xrightarrow{\text{speed of light}} = 5 \times 10^6 \text{ bits}$$

c) Bit width of data in link = $S = \frac{2 \times 10^8}{R} = \frac{2 \times 10^8}{100 \times 10^9} = 2 \times 10^{-3} \text{ m}$

4) Let the file transfer time be = $k \text{ sec/byte}$.

a) Non persistent : $2RTT + 1000k + \sum_{i=1}^{\infty} 2RTT + (100 \times 10^3)^3 k$

Setting up connection + requesting webpage. = $2RTT + k[10^6 + 10^3]$

$$= 0.22 + k[10^6 + 10^3] \text{ s.}$$

b) Persistent connection : Connection needn't be setup again.

$$2RTT + \frac{10^3 k}{k} + \sum_{i=1}^{10} (RTT + \frac{10^5 k}{k}) s$$

$$= 12RTT + k(10^6 + 10^3)$$

$$= 0.12 + k(10^6 + 10^3) s$$

c) Persistent + pipelined & dataframes of 1 kB each.

all images will be packed & sent in 1 RTT

$$2RTT + \frac{10^3 k}{k} + RTT + \sum_{i=1}^{10} (10^5 k)$$

$$= 3RTT + k(10^3 + 10^6)$$

$$= 0.03 + k(10^3 + 10^6) s$$

Part3: Answer 5:

- a) The new protocols are:
- i) TLSv1.3 [Transport Layer Security Version 1.3]: It provides data encryption services and secures communication channels between two endpoints. It aims to encrypt handshakes to the maximum possible extent. It's the latest version of the internet's most deployed security channel with better security than previous versions and eliminates the usage of obsolete cryptographic algorithms. It is located between the application protocol layer and TCP/IP layer. Details about this protocol can be found in RFC 8446.
 - ii) QUIC [Quick UDP Internet Connections]: It's an experimental secure internet transport protocol designed by Google, similar to TCP+TLS+HTTP/2 implemented on UDP, with an aim to reduce the latency in comparison with TCP. It's a transport layer protocol whose details can be found in RFC 9000.
 - iii) MDNS [Multicast DNS]: It helps in resolution of names on smaller networks without using pre configured server [DNS] like in cases when local DNS doesn't exist. nDNS requests and responses are UDP with port 5353 as both source and destination ports. It is an application layer protocol with details present in RFC 6762.
 - iv) ICMPv6 [Internet Control Message Protocol]: It is Internet Control Message Protocol for IP version 6 and performs diagnostic and error reporting functions. It is layer 3 [Network Layer] protocol and details about it are present in RFC 4443.
 - v) NTP [Network Time Protocol]: It is an Application Layer protocol that helps in clock synchronization [aims to coordinate within a few milliseconds of UTC] between computer systems with variable latency data networks following packet switching. Details about its current version can be found in RFC 5905.
 - vi) OCSP [Online Certificate Status Protocol]: It is used to check digital certificate's validity. Browsers like FireFox use OCSP to validate HTTP certificates. It is Transport layer Internet Protocol with details in RFC 6960.

b) RTT for connection established by pinging yahoo.com (ping yahoo.com -c 1).

$$\text{RTT} = \text{Time(reply } \{46\}) - \text{Time(request } \{45\}) = 10.428544546 - 10.129917005 = 298.6 \text{ ms.}$$

No.	Time	Source	Destination	Protocol	Length	Info
43	10.089955546	192.168.64.1	192.168.64.2	DNS	176	Standard query response 0x28bf A yahoo.com
44	10.128235838	192.168.64.1	192.168.64.2	DNS	248	Standard query response 0x6bb5 AAAA yahoo.com
45	10.129917005	192.168.64.2	74.6.143.26	ICMP	98	Echo (ping) request id=0x0001, seq=1/256,
46	10.428544546	74.6.143.26	192.168.64.2	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256,
47	10.429446171	192.168.64.2	192.168.64.1	DNS	95	Standard query 0xa57b PTR 26.143.6.74.in-a
48	10.756566130	192.168.64.1	192.168.64.2	DNS	155	Standard query response 0xa57b PTR 26.143.
49	11.409459172	192.168.64.2	142.250.183.202	UDP	71	57170 → 443 Len=29
50	11.433878880	142.250.183.202	192.168.64.2	UDP	70	443 → 57170 Len=28
51	17.392447783	142.250.183.202	192.168.64.2	UDP	124	443 → 57170 Len=82

c)

As soon as I visited ims.iitgn.ac.in, 2 cookies were set up (_ga, _gid) and a third one was set up when I logged into the student portal (RequestToken). Cookies are stored as name-value pairs (specified in the corresponding headers below). Expiry date and time of the first 2 cookies are given under their respective “Expires” tab. The third cookie is a Session cookie i.e. it will expire as soon as the session is closed. Domain attribute specifies hosts that receive a cookie. Path specifies the webpage or the directory that sets up the cookie. The size of the cookie is specified under the size field. The sizes of cookie 1,2 and 3 are 30B, 31B and 36B respectively. Secure field checked implies the cookie is sent over HTTPS only with an encrypted request and whether the cookie can be retrieved by any server. The first 2 cookies are secure while the third one isn't. HttpOnly (checked) prevents client's access to cookie's data. SameSite Lax implies a more relaxed form of protection of cross-site requests.

Name	Value	Domain	Path	Expires	Size	Secure	HttpOnly	SameSite
_ga	GA1.1466809802.1663140408	.iitgn.ac.in	/	21/9/2022, 12:56:47 PM	30 B	✓	✓	-
_gid	GA1.3.1696849012.1663140408	.iitgn.ac.in	/	15/9/2022, 12:56:47 PM	31 B	✓	✓	-
RequestToken	aeqjyweopwi3wroojg4iemt	ims.iitgn.ac.in	/	Session	36 B	✓	✓	Lax