Joy Makwana (20110086)

DESIGN DOCUMENT

I have used three modes of layering:

- **File Service:** The top layer is the file service layer which ensures the implementation of RFS commands. It uses the OS APIs to fetch various information from server and client; some of the APIs are used for fetching the current working directory, changing the directory, and also for fetching the list of folders and files. This layer relies on the crypto service for encryption/ decryption or encoding/ decoding of the data on the server end and client end.

- **Crypto Service:** The middle layer is a crypto layer that facilitates the encryption of the data which is being transferred from the socket. The encryption of the data is mainly implemented in three ways: Plain Text, Substitution, and Transpose. These encryption methods mangles the data before it gets encoded and sent through the socket.

    - Plain Text: In plain text, there is no need to mangle the data, i.e., we just have to encode the data if it's a string and send it otherwise, if it's in bytes, then we just have to send the data without even encoding.

    - Substitution: In substitution, we substitute only the alphanumeric characters with an offset of 2, or we can say that the caesar cipher with an offset of 2. For implementing this, I have used two functions called "sub" and "desub", the sub is used to substitute the alphanumeric characters with an offset of 2, and desub is used to substitute alphanumeric characters with a decrement of 2. We use sub when we send the data and then encode it, at the receiving end, we decode the data first and then use desub to get the correct data from the mangled data.

- Transpose: In transpose, I am mangling the data by reversing it word by word. For implementing transpose, I made a function called "transpose," which takes data strings as a parameter and then it just reverses each word of the string by keeping space between the words in consideration. For using transpose, we first mangle the data with the transpose function and then encode it and then send it, while we decode it first at the receiving end and then again pass the data string into the same transpose function to get the correct data back.

- Networking service: The bottom layer is a networking layer. It facilitates the protocol for transferring data. I have used TCP for transferring data between the client and server as it can be easily built using python's in-built library called "socket," which helps us to create a socket between the client and server. After creating the socket, I easily transferred the data between the client and server. For using the socket, I bound one end of the socket to the server and then connected the other end to the client. Then I gave a port to the server and an IP address to the client, which helps the socket to differentiate between multiple clients but in this assignment, I have implemented a single client system.

Associated challenges were:

- I first implemented the sockets by using C language but later realized that in C language, many libraries which I used are different for Linux and windows. So, I would have to write different code for using it on linux. Also, the process of creating sockets in C was lengthy. Therefore, I switched to python.

- The next problem I faced was in the OS library. I implemented CD, and CWD commands easily, but for LS, I was getting a list instead of a string, and I can't directly just send a list. So, I converted the list into the string, then encoded it and sent it to the client.

- When I was implementing DWD and UPD commands, I used a “for loop” in Plain Text encryption, which was working. But, when I implemented the same logic in transpose encryption then, I couldn’t get the correct data after downloading the file then I realized that in the “for loop,” the size of data sent is not defined, and at the receiver’s end, I defined the size as 1024 bytes due to which some data might get lost or corrupted in the process. So, I used a “while loop” and defined the data size to be sent instead of using for loop, and it worked.

- Currently, my code can download only txt files for substitution and transpose modes of encryption, but I can send jpg, png, and txt files for plain text. I tried to implement sharing png files in all the modes, but I didn’t succeed. In transpose, I can share png if I consider 8 bytes as a word and transfer 8 bytes at a time by reversing them. Later, I implemented transpose by sending 1024 bytes at a time and a considered string as a word when there is a space between two strings, but in this case, I can only download text files. So, I can only send and receive text files for this assignment.

Screenshots:

<u>PlainText:</u>

- CWD:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\makwa\CN> cd client
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: PlainText
[client]: cwd
[server]: C:\Users\makwa\CN\server
[client]:
```

- CD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: PlainText
[client]: cd f1
[server]: Directory has been changed to f1
[client]: cwd
[server]: C:\Users\makwa\CN\server\f1
[client]: cd f2
[server]: Directory has been changed to f2
[client]: cwd
[server]: C:\Users\makwa\CN\server\f1\f2
```

- LS:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: PlainText
[client]: ls
[server]: ['architec.png', 'f1', 'f1.txt', 'function.py', 'server.py', '__pycache__']
[client]:
```

- DWD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: PlainText
[client]: dwd txt1.txt
[server]: Downloaded!
PS C:\Users\makwa\CN\client>
```

- UPD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: PlainText
[client]: upd txt2.txt
File uploaded!
PS C:\Users\makwa\CN\client>
```

Substitution:

- CWD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Substitution
[client]: cwd
[server]: C:\Users\makwa\CN\server
[client]:
```

- CD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Substitution
[client]: cd f1
[server]: Directory has been changed to f1
[client]: cwd
[server]: C:\Users\makwa\CN\server\f1
[client]:
```

- LS:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Substitution
[client]: ls
[server]: ['architec.png', 'f1', 'function.py', 'server.py', 'txt1.txt', 'txt2.txt', '__pycache__']
[client]:
```

- DWD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Substitution
[client]: dwd txt.txt
[server]: Downloaded!
PS C:\Users\makwa\CN\client>
```

- UPD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Substitution
[client]: upd txt.txt
File uploaded!
PS C:\Users\makwa\CN\client>
```

Transpose:

- CWD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Transpose
[client]: cwd
[server]: C:\Users\makwa\CN\server
[client]:
```

- CD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Transpose
[client]: cd f1
[server]: Directory has been changed to f1
[client]: cwd
[server]: C:\Users\makwa\CN\server\f1
```

- LS:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Transpose
[client]: ls
[server]: ['architec.png', 'f1', 'function.py', 'server.py', 'txt.txt', 'txt2.txt', '__pycache__']
[client]:
```

- DWD:

```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Transpose
[client]: ls
[server]: ['architec.png', 'f1', 'function.py', 'server.py', 'txt.txt', 'txt2.txt', '__pycache__']
[client]: dwd txt2.txt
[server]: Downloaded!
PS C:\Users\makwa\CN\client>
```

- UPD:

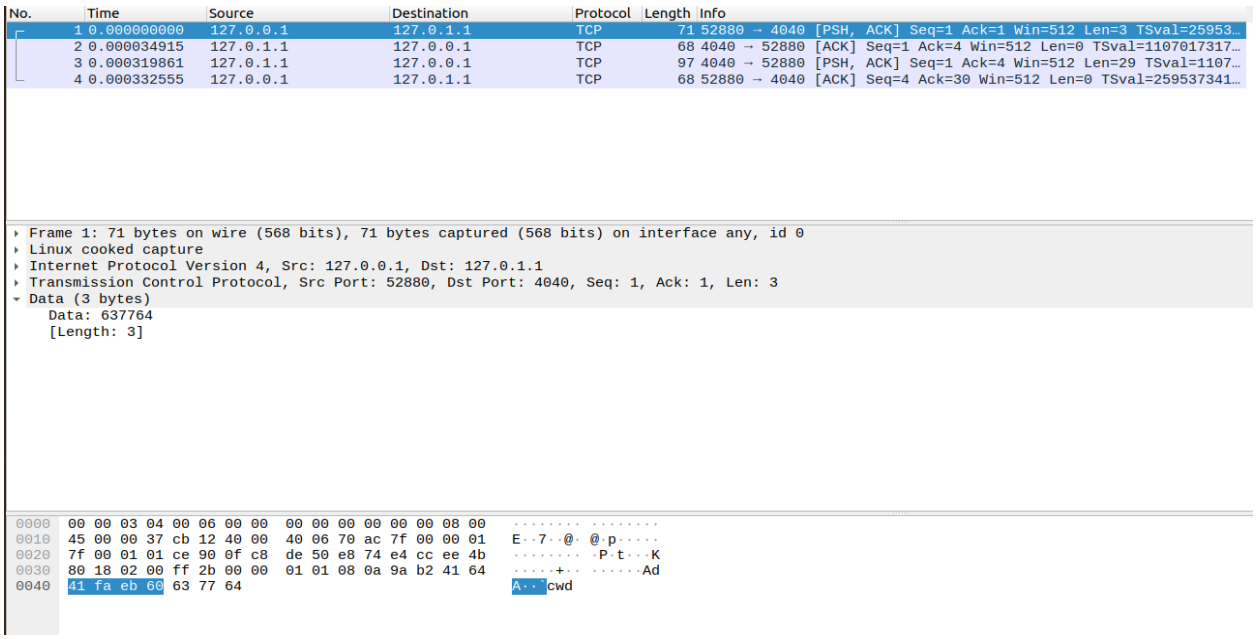
```
PS C:\Users\makwa\CN\client> py client.py
[client]: Select any one method of encryption from the following: PlainText, Substitution, Transpose: Transpose
[client]: upd txt.txt
[server]: File uploaded!
PS C:\Users\makwa\CN\client>
```

(C) Wireshark Analysis

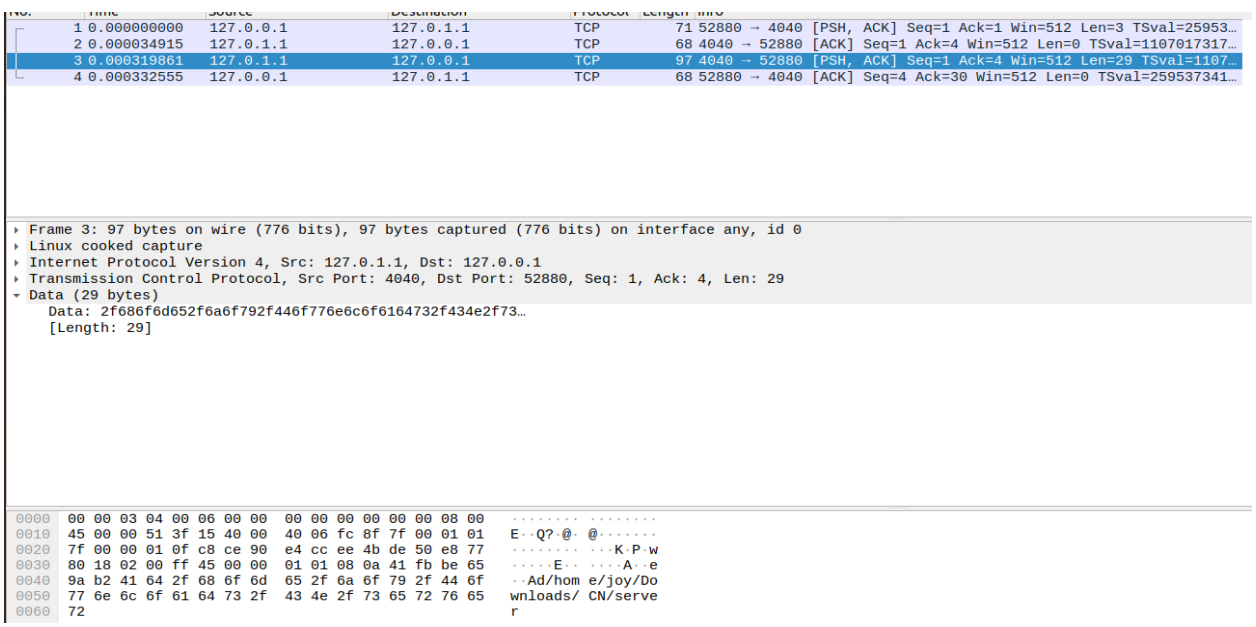
I have used three modes of encryption, and the encrypted data for each mode is:

- Plain Text:

CWD :



Client is sending the message "cwd" in plain text encryption



Server sent the path of the current working directory on the command "cwd"

Ls:

No.	Time	Source	Destination	Protocol	Length	Info
3	2.202307082	127.0.0.1	127.0.1.1	TCP	70	52880 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=2 TSval=25956…
4	2.202334943	127.0.1.1	127.0.0.1	TCP	68	4040 → 52880 [ACK] Seq=1 Ack=3 Win=512 Len=0 TSval=1107308828…
5	2.202713747	127.0.1.1	127.0.0.1	TCP	150	4040 → 52880 [PSH, ACK] Seq=1 Ack=3 Win=512 Len=82 TSval=1107…
6	2.202792250	127.0.0.1	127.0.1.1	TCP	68	52880 → 4040 [ACK] Seq=3 Ack=83 Win=512 Len=0 TSval=259566492…

```
▸ Frame 3: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52880, Dst Port: 4040, Seq: 1, Ack: 1, Len: 2
▾ Data (2 bytes)
    Data: 6c73
    [Length: 2]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 36 cb 14 40 00  40 06 70 ab 7f 00 00 01   E··6··@· @·p·····
0020  7f 00 01 01 ce 90 0f c8  de 50 e8 77 e4 cc ee 68   ........ ·P·w···h
0030  80 18 02 00 ff 2a 00 00  01 01 08 0a 9a b6 b4 1b   ·····*·· ........
0040  41 fb be 65 6c 73                                  A··els
```

Client is sending “ls” command to the server

No.	Time	Source	Destination	Protocol	Length	Info
3	2.202307082	127.0.0.1	127.0.1.1	TCP	70	52880 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=2 TSval=25956…
4	2.202334943	127.0.1.1	127.0.0.1	TCP	68	4040 → 52880 [ACK] Seq=1 Ack=3 Win=512 Len=0 TSval=1107308828…
5	2.202713747	127.0.1.1	127.0.0.1	TCP	150	4040 → 52880 [PSH, ACK] Seq=1 Ack=3 Win=512 Len=82 TSval=1107…
6	2.202792250	127.0.0.1	127.0.1.1	TCP	68	52880 → 4040 [ACK] Seq=3 Ack=83 Win=512 Len=0 TSval=259566492…

```
▸ Frame 5: 150 bytes on wire (1200 bits), 150 bytes captured (1200 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52880, Seq: 1, Ack: 3, Len: 82
▾ Data (82 bytes)
    Data: 5b277478742e747874272c202766756e6374696f6e2e7079…
    [Length: 82]
```

```
0010  45 00 00 86 3f 17 40 00  40 06 fc 58 7f 00 01 01   E···?·@· @··X····
0020  7f 00 00 01 0f c8 ce 90  e4 cc ee 68 de 50 e8 79   ........ ···h·P·y
0030  80 18 02 00 ff 7a 00 00  01 01 08 0a 42 00 31 1d   ·····z·· ····B·1·
0040  9a b6 b4 1b 5b 27 74 78  74 2e 74 78 74 27 2c 20   ····['tx t.txt',
0050  27 66 75 6e 63 74 69 6f  6e 2e 70 79 27 2c 20 27   'functio n.py', '
0060  73 65 72 76 65 72 2e 70  79 27 2c 20 27 74 78 74   server.p y', 'txt
0070  32 2e 74 78 74 27 2c 20  27 61 72 63 68 69 74 65   2.txt',  'archite
0080  63 2e 70 6e 67 27 2c 20  27 5f 5f 70 79 63 61 63   c.png',  '__pycac
0090  68 65 5f 5f 27 5d                                  he__']
```

Server sent the string containing all the files and folders in the cwd

cd:

No.	Time	Source	Destination	Protocol	Length	Info
13	126.525188629	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
14	126.525245672	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
15	126.525403083	10.0.2.15	34.122.121.32	HTTP	143	GET / HTTP/1.1
16	126.525635224	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [ACK] Seq=1 Ack=88 Win=65535 Len=0
19	137.748720014	34.122.121.32	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
20	137.748765185	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0
21	137.748720507	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [FIN, ACK] Seq=149 Ack=88 Win=65535 Len=0
22	137.749217536	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [FIN, ACK] Seq=88 Ack=150 Win=64091 Len=0
23	137.749634600	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [ACK] Seq=150 Ack=89 Win=65535 Len=0
24	149.727967232	127.0.0.1	127.0.1.1	TCP	73	52880 → 4040 [PSH, ACK] Seq=3 Ack=83 Win=512 Len=5 TSval=2595…
25	149.728135623	127.0.1.1	127.0.0.1	TCP	100	4040 → 52880 [PSH, ACK] Seq=83 Ack=8 Win=512 Len=32 TSval=110…
26	149.728154083	127.0.0.1	127.0.1.1	TCP	68	52880 → 4040 [ACK] Seq=8 Ack=115 Win=512 Len=0 TSval=25958124…

```
▸ Frame 24: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52880, Dst Port: 4040, Seq: 3, Ack: 83, Len: 5
▾ Data (5 bytes)
    Data: 6364206631
    [Length: 5]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 39 cb 16 40 00  40 06 70 a6 7f 00 00 01   E··9··@· @·p·····
0020  7f 00 01 01 ce 90 0f c8  de 50 e8 79 e4 cc ee ba   ........ ·P·y····
0030  80 18 02 00 ff 2d 00 00  01 01 08 0a 9a b8 f4 61   ·····-·· ·******a
0040  42 00 31 1d 63 64 20 66  31                        B·1·cd f 1
```

Client is sending command “cd f1” to the server

No.	Time	Source	Destination	Protocol	Length	Info
13	126.525188629	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
14	126.525245672	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
15	126.525403083	10.0.2.15	34.122.121.32	HTTP	143	GET / HTTP/1.1
16	126.525635224	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [ACK] Seq=1 Ack=88 Win=65535 Len=0
19	137.748720014	34.122.121.32	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
20	137.748765185	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0
21	137.748720507	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [FIN, ACK] Seq=149 Ack=88 Win=65535 Len=0
22	137.749217536	10.0.2.15	34.122.121.32	TCP	56	40232 → 80 [FIN, ACK] Seq=88 Ack=150 Win=64091 Len=0
23	137.749634600	34.122.121.32	10.0.2.15	TCP	62	80 → 40232 [ACK] Seq=150 Ack=89 Win=65535 Len=0
24	149.727967232	127.0.0.1	127.0.1.1	TCP	73	52880 → 4040 [PSH, ACK] Seq=3 Ack=83 Win=512 Len=5 TSval=2595…
25	149.728135623	127.0.1.1	127.0.0.1	TCP	100	4040 → 52880 [PSH, ACK] Seq=83 Ack=8 Win=512 Len=32 TSval=110…
26	149.728154083	127.0.0.1	127.0.1.1	TCP	68	52880 → 4040 [ACK] Seq=8 Ack=115 Win=512 Len=0 TSval=25958124…

```
▸ Frame 25: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52880, Seq: 83, Ack: 8, Len: 32
▾ Data (32 bytes)
    Data: 4469726563746f72792068617320626565 6e206368616e67…
    [Length: 32]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ********
0010  45 00 00 54 3f 18 40 00  40 06 fc 89 7f 00 01 01   E··T?·@· @*******
0020  7f 00 00 01 0f c8 ce 90  e4 cc ee ba de 50 e8 7e   ******** *****P·~
0030  80 18 02 00 ff 48 00 00  01 01 08 0a 42 02 71 62   *****H·· ****B·qb
0040  9a b8 f4 61 44 69 72 65  63 74 6f 72 79 20 68 61   ···aDire ctory ha
0050  73 20 62 65 65 6e 20 63  68 61 6e 67 65 64 20 74   s been c hanged t
0060  6f 20 66 31                                        o f1
```

Server changed the cwd to f1 as asked by the client and server informed the client

Dwd and Upd:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	74	52910 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=6 TSval=25999…
2	0.000028213	127.0.1.1	127.0.0.1	TCP	68	4040 → 52910 [ACK] Seq=1 Ack=7 Win=512 Len=0 TSval=1111583779…
3	0.000379504	127.0.1.1	127.0.0.1	TCP	81	4040 → 52910 [PSH, ACK] Seq=1 Ack=7 Win=512 Len=13 TSval=1111…
4	0.000392330	127.0.0.1	127.0.1.1	TCP	68	52910 → 4040 [ACK] Seq=7 Ack=14 Win=512 Len=0 TSval=259993987…
5	0.000445631	127.0.1.1	127.0.0.1	TCP	79	4040 → 52910 [PSH, ACK] Seq=14 Ack=7 Win=512 Len=11 TSval=111…
6	0.000451306	127.0.0.1	127.0.1.1	TCP	68	52910 → 4040 [ACK] Seq=7 Ack=25 Win=512 Len=0 TSval=259993987…
7	0.000479989	127.0.1.1	127.0.0.1	TCP	68	4040 → 52910 [FIN, ACK] Seq=25 Ack=7 Win=512 Len=0 TSval=1111…
8	0.001067742	127.0.0.1	127.0.1.1	TCP	68	52910 → 4040 [FIN, ACK] Seq=7 Ack=26 Win=512 Len=0 TSval=2599…
9	0.001086180	127.0.1.1	127.0.0.1	TCP	68	4040 → 52910 [ACK] Seq=26 Ack=8 Win=512 Len=0 TSval=111158378…

```
▸ Frame 3: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52910, Seq: 1, Ack: 7, Len: 13
▸ Data (13 bytes)
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 41 00 71 40 00  40 06 3b 44 7f 00 01 01   E··A·q@· @·;D····
0020  7f 00 00 01 0f c8 ce ae  31 02 86 dd 90 a4 fa f2   ........ 1·......
0030  80 18 02 00 ff 35 00 00  01 01 08 0a 42 41 6c 24   ·····5·· ····BAl$
0040  9a f7 ef 22 48 65 6c 6c  6f 20 57 6f 72 6c 64 21   ···"Hell o World!
0050  0a                                                 ·
```

Server sent the data “hello world” which gets written in the file created in the client. In upload, client will sent the data and server will write it.

Substitution:

Cwd:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	71	52886 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=3 TSval=25962…
2	0.000035185	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=1 Ack=4 Win=512 Len=0 TSval=1107869228…
3	0.000242092	127.0.1.1	127.0.0.1	TCP	97	4040 → 52886 [PSH, ACK] Seq=1 Ack=4 Win=512 Len=29 TSval=1107…
4	0.000256270	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=4 Ack=30 Win=512 Len=0 TSval=259622532…

```
▸ Frame 1: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface any, id 0
▸ Linux cooked capture
▾ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 55
    Identification: 0x03e2 (994)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x37dd [validation disabled]
    [Header checksum status: Unverified]
    Source: 127.0.0.1
    Destination: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52886, Dst Port: 4040, Seq: 1, Ack: 1, Len: 3
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 37 03 e2 40 00  40 06 37 dd 7f 00 00 01   E··7··@· @·7·····
0020  7f 00 01 01 ce 96 0f c8  3b d8 f6 72 96 96 97 6d   ........ ;··r···m
0030  80 18 02 00 ff 2b 00 00  01 01 08 0a 9a bf 41 2b   ·····+·· ......A+
0040  42 07 df 4a 65 79 66                               B··Jeyf
```

Client send the command “cwd” to the server which got encrypted as “eyf” (offset=2)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	71	52886 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=3 TSval=25962…
2	0.000035185	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=1 Ack=4 Win=512 Len=0 TSval=1107869228…
3	0.000242092	127.0.1.1	127.0.0.1	TCP	97	4040 → 52886 [PSH, ACK] Seq=1 Ack=4 Win=512 Len=29 TSval=1107…
4	0.000256270	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=4 Ack=30 Win=512 Len=0 TSval=259622532…

```
▸ Frame 3: 97 bytes on wire (776 bits), 97 bytes captured (776 bits) on interface any, id 0
▸ Linux cooked capture
▾ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 81
    Identification: 0x691e (26910)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0xd286 [validation disabled]
    [Header checksum status: Unverified]
    Source: 127.0.1.1
    Destination: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52886, Seq: 1, Ack: 4, Len: 29

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 51 69 1e 40 00  40 06 d2 86 7f 00 01 01   E··Qi·@· @·......
0020  7f 00 00 01 0f c8 ce 96  96 96 97 6d 3b d8 f6 75   ........ ···m;··u
0030  80 18 02 00 ff 45 00 00  01 01 08 0a 42 08 be 2c   ·····E·· ····B··,
0040  9a bf 41 2b 2f 6a 71 6f  67 2f 6c 71 61 2f 46 71   ··A+/jqo g/lqa/Fq
0050  79 70 6e 71 63 66 75 2f  45 50 2f 75 67 74 78 67   ypnqcfu/ EP/ugtxg
0060  74                                                 t
```

Server sent the corresponding answer in the encrypted form with an offset of 2

Ls:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	71	52886 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=3 TSval=25962…
2	0.000035185	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=1 Ack=4 Win=512 Len=0 TSval=1107869228…
3	0.000242092	127.0.1.1	127.0.0.1	TCP	97	4040 → 52886 [PSH, ACK] Seq=1 Ack=4 Win=512 Len=29 TSval=1107…
4	0.000256270	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=4 Ack=30 Win=512 Len=0 TSval=259622532…
13	112.168366046	127.0.0.1	127.0.1.1	TCP	70	52886 → 4040 [PSH, ACK] Seq=4 Ack=30 Win=512 Len=2 TSval=2596…
14	112.168397525	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=30 Ack=6 Win=512 Len=0 TSval=110798139…
15	112.168978265	127.0.1.1	127.0.0.1	TCP	156	4040 → 52886 [PSH, ACK] Seq=30 Ack=6 Win=512 Len=88 TSval=110…
16	112.168990287	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=6 Ack=118 Win=512 Len=0 TSval=25963374…

```
▸ Frame 13: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface any, id 0
▸ Linux cooked capture
▾ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 54
    Identification: 0x03e4 (996)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x37dc [validation disabled]
    [Header checksum status: Unverified]
    Source: 127.0.0.1
    Destination: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52886, Dst Port: 4040, Seq: 4, Ack: 30, Len: 2

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 36 03 e4 40 00  40 06 37 dc 7f 00 00 01   E··6··@· @·7·····
0020  7f 00 01 01 ce 96 0f c8  3b d8 f6 75 96 96 97 8a   ........ ;··u····
0030  80 18 02 00 ff 2a 00 00  01 01 08 0a 9a c0 f7 53   ·····*·· ·******S
0040  42 08 be 2c 6e 75                                  B··,nu
```

Client sent the encrypted message “nu” which means “ls”

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	71	52886 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=3 TSval=25962…
2	0.000035185	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=1 Ack=4 Win=512 Len=0 TSval=1107869228…
3	0.000242092	127.0.1.1	127.0.0.1	TCP	97	4040 → 52886 [PSH, ACK] Seq=1 Ack=4 Win=512 Len=29 TSval=1107…
4	0.000256270	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=4 Ack=30 Win=512 Len=0 TSval=259622532…
13	112.168366046	127.0.0.1	127.0.1.1	TCP	70	52886 → 4040 [PSH, ACK] Seq=4 Ack=30 Win=512 Len=2 TSval=2596…
14	112.168397525	127.0.1.1	127.0.0.1	TCP	68	4040 → 52886 [ACK] Seq=30 Ack=6 Win=512 Len=0 TSval=110798139…
15	112.168978265	127.0.1.1	127.0.0.1	TCP	156	4040 → 52886 [PSH, ACK] Seq=30 Ack=6 Win=512 Len=88 TSval=110…
16	112.168990287	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=6 Ack=118 Win=512 Len=0 TSval=25963374…

```
▸ Frame 15: 156 bytes on wire (1248 bits), 156 bytes captured (1248 bits) on interface any, id 0
▸ Linux cooked capture
▾ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 140
    Identification: 0x6920 (26912)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0xd249 [validation disabled]
    [Header checksum status: Unverified]
    Source: 127.0.1.1
    Destination: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52886, Seq: 30, Ack: 6, Len: 88
```

```
0010  45 00 00 8c 69 20 40 00  40 06 d2 49 7f 00 01 01   E···i @· @··I····
0020  7f 00 00 01 0f c8 ce 96  96 96 97 8a 3b d8 f6 77   ........ ····;··w
0030  80 18 02 00 ff 80 00 00  01 01 08 0a 42 0a 74 55   ........ ····B·tU
0040  9a c0 f7 53 5b 27 76 7a  76 2e 76 7a 76 27 2c 20   ···S['vz v.vzv',
0050  27 68 77 70 65 76 6b 71  70 2e 72 61 27 2c 20 27   'hwpevkq p.ra', '
0060  68 33 27 2c 20 27 75 67  74 78 67 74 2e 72 61 27   h3', 'ug txgt.ra'
0070  2c 20 27 76 7a 76 34 2e  76 7a 76 27 2c 20 27 63   , 'vzv4. vzv', 'c
0080  74 65 6a 6b 76 67 65 2e  72 70 69 27 2c 20 27 5f   tejkvge. rpi', '_
0090  5f 72 61 65 63 65 6a 67  5f 5f 27 5d               _raecejg __']
```

Server sent the corresponding answer in the encrypted form with an offset of 2

Cd:

No.	Time	Source	Destination	Protocol	Length	Info
49	186.835115580	10.0.2.15	35.224.170.84	HTTP	143	GET / HTTP/1.1
50	186.835706970	35.224.170.84	10.0.2.15	TCP	62	80 → 49400 [ACK] Seq=1 Ack=88 Win=65535 Len=0
51	190.827434295	10.0.2.15	35.224.170.84	TCP	76	49402 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
52	190.831182633	35.224.170.84	10.0.2.15	TCP	62	80 → 49402 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
53	190.831215799	10.0.2.15	35.224.170.84	TCP	56	49402 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
54	190.831411560	10.0.2.15	35.224.170.84	HTTP	143	GET / HTTP/1.1
55	190.831846135	35.224.170.84	10.0.2.15	TCP	62	80 → 49402 [ACK] Seq=1 Ack=88 Win=65535 Len=0
56	197.872478631	127.0.0.1	127.0.1.1	TCP	73	52886 → 4040 [PSH, ACK] Seq=6 Ack=118 Win=512 Len=5 TSval=259…
57	197.872691509	127.0.1.1	127.0.0.1	TCP	100	4040 → 52886 [PSH, ACK] Seq=118 Ack=11 Win=512 Len=32 TSval=1…
58	197.872699936	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=11 Ack=150 Win=512 Len=0 TSval=2596423…
59	198.006257890	35.224.170.84	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
60	198.006320210	10.0.2.15	35.224.170.84	TCP	56	49398 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0

```
▸ Frame 56: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52886, Dst Port: 4040, Seq: 6, Ack: 118, Len: 5
▾ Data (5 bytes)
    Data: 6566206833
    [Length: 5]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 39 03 e6 40 00  40 06 37 d7 7f 00 00 01   E··9··@· @·7·····
0020  7f 00 01 01 ce 96 0f c8  3b d8 f6 77 96 96 97 e2   ........ ;··w····
0030  80 18 02 00 ff 2d 00 00  01 01 08 0a 9a c2 46 1b   ·····-·· ......F·
0040  42 0a 74 55 65 66 20 68  33                        B·tUef h 3
```

Client sent the encrypted message “ef h3” which means “cd f1”

No.	Time	Source	Destination	Protocol	Length	Info
49	186.835115580	10.0.2.15	35.224.170.84	HTTP	143	GET / HTTP/1.1
50	186.835706970	35.224.170.84	10.0.2.15	TCP	62	80 → 49400 [ACK] Seq=1 Ack=88 Win=65535 Len=0
51	190.827434295	10.0.2.15	35.224.170.84	TCP	76	49402 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
52	190.831182633	35.224.170.84	10.0.2.15	TCP	62	80 → 49402 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
53	190.831215799	10.0.2.15	35.224.170.84	TCP	56	49402 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
54	190.831411560	10.0.2.15	35.224.170.84	HTTP	143	GET / HTTP/1.1
55	190.831846135	35.224.170.84	10.0.2.15	TCP	62	80 → 49402 [ACK] Seq=1 Ack=88 Win=65535 Len=0
56	197.872478631	127.0.0.1	127.0.1.1	TCP	73	52886 → 4040 [PSH, ACK] Seq=6 Ack=118 Win=512 Len=5 TSval=259…
57	197.872691509	127.0.1.1	127.0.0.1	TCP	100	4040 → 52886 [PSH, ACK] Seq=118 Ack=11 Win=512 Len=32 TSval=1…
58	197.872699936	127.0.0.1	127.0.1.1	TCP	68	52886 → 4040 [ACK] Seq=11 Ack=150 Win=512 Len=0 TSval=2596423…
59	198.006257890	35.224.170.84	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
60	198.006320210	10.0.2.15	35.224.170.84	TCP	56	49398 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0

```
▸ Frame 57: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52886, Seq: 118, Ack: 11, Len: 32
▾ Data (32 bytes)
    Data: 466b74676576717461206a637520646767702065 6a637069…
    [Length: 32]

0000  00 00 03 04 00 06 00 00  00 00 00 00 d2 a2 08 00   ........ ........
0010  45 00 00 54 69 21 40 00  40 06 d2 80 7f 00 01 01   E··Ti!@· @·......
0020  7f 00 00 01 0f c8 ce 96  96 96 97 e2 3b d8 f6 7c   ........ ····;··|
0030  80 18 02 00 ff 48 00 00  01 01 08 0a 42 0b c3 1c   ·····H·· ····B···
0040  9a c2 46 1b 46 6b 74 67  65 76 71 74 61 20 6a 63   ··F·Fktg evqta jc
0050  75 20 64 67 67 70 20 65  6a 63 70 69 67 66 20 76   u dggp e jcpigf v
0060  71 20 68 33                                        q h3
```

Server sent the message in the encrypted form with an offset of 2 which decrypts to "Directory has been changed to f1"

DWD and UPD:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	74	52924 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=6 TSval=26018…
2	0.000018518	127.0.1.1	127.0.0.1	TCP	68	4040 → 52924 [ACK] Seq=1 Ack=7 Win=512 Len=0 TSval=1113466920…
3	0.000514134	127.0.1.1	127.0.0.1	TCP	81	4040 → 52924 [PSH, ACK] Seq=1 Ack=7 Win=512 Len=13 TSval=1113…
4	0.000527583	127.0.0.1	127.0.1.1	TCP	68	52924 → 4040 [ACK] Seq=7 Ack=14 Win=512 Len=0 TSval=260182301…
5	0.000649743	127.0.1.1	127.0.0.1	TCP	79	4040 → 52924 [PSH, ACK] Seq=14 Ack=7 Win=512 Len=11 TSval=111…
6	0.000656625	127.0.0.1	127.0.1.1	TCP	68	52924 → 4040 [ACK] Seq=7 Ack=25 Win=512 Len=0 TSval=260182301…
7	0.000684569	127.0.1.1	127.0.0.1	TCP	68	4040 → 52924 [FIN, ACK] Seq=25 Ack=7 Win=512 Len=0 TSval=1113…
8	0.000819239	127.0.0.1	127.0.1.1	TCP	68	52924 → 4040 [FIN, ACK] Seq=7 Ack=26 Win=512 Len=0 TSval=2601…
9	0.000836324	127.0.1.1	127.0.0.1	TCP	68	4040 → 52924 [ACK] Seq=26 Ack=8 Win=512 Len=0 TSval=111346692…

```
▸ Frame 3: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52924, Seq: 1, Ack: 7, Len: 13
▸ Data (13 bytes)

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 41 49 05 40 00  40 06 f2 af 7f 00 01 01   E··AI·@· @·......
0020  7f 00 00 01 0f c8 ce bc  ef 36 49 21 d0 f2 b5 9b   ........ ·6I!····
0030  80 18 02 00 ff 35 00 00  01 01 08 0a 42 5e 28 29   ·····5·· ····B^()
0040  9b 14 ab 27 4a 67 6e 6e  71 20 59 71 74 6e 66 21   ···'Jgnn q Yqtnf!
0050  0a                                                 ·
```

Data will get encrypted and sent in download and upload in the same but the sender and receiver will change. For dwd, sender is server and for upd, receiver is server.

Transpose:

Cwd:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	72	52888 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=4 TSval=25966
2	0.000014846	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=1 Ack=5 Win=512 Len=0 TSval=1108329186
3	0.000186553	127.0.1.1	127.0.0.1	TCP	98	4040 → 52888 [PSH, ACK] Seq=1 Ack=5 Win=512 Len=30 TSval=1108
4	0.000192834	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=5 Ack=31 Win=512 Len=0 TSval=259668528

```
▸ Frame 1: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52888, Dst Port: 4040, Seq: 1, Ack: 1, Len: 4
▾ Data (4 bytes)
    Data: 64776320
    [Length: 4]

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 38 85 f5 40 00  40 06 b5 c8 7f 00 00 01   E··8··@· @·......
0020  7f 00 01 01 ce 98 0f c8  87 34 2c d3 7c 51 ba 78   ........ ·4,·|Q·x
0030  80 18 02 00 ff 2c 00 00  01 01 08 0a 9a c6 45 e1   ·····,·· ......E·
0040  42 0f 9f 1f 64 77 63 20                             B···dwc
```

client sent the message in encrypted format which means it reversed “cwd” to “dwc”

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	TCP	72	52888 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=4 TSval=25966…
2	0.000014846	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=1 Ack=5 Win=512 Len=0 TSval=1108329186…
3	0.000186553	127.0.1.1	127.0.0.1	TCP	98	4040 → 52888 [PSH, ACK] Seq=1 Ack=5 Win=512 Len=30 TSval=1108…
4	0.000192834	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=5 Ack=31 Win=512 Len=0 TSval=259668528…

```
▸ Frame 3: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52888, Seq: 1, Ack: 5, Len: 30
▾ Data (30 bytes)
    Data: 7265767265732f4e432f7364616f6c6e776f442f796f6a2f…
    [Length: 30]

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 52 89 72 40 00  40 06 b2 31 7f 00 01 01   E··R·r@· @··1····
0020  7f 00 00 01 0f c8 ce 98  7c 51 ba 78 87 34 2c d7   ........ |Q·x·4,·
0030  80 18 02 00 ff 46 00 00  01 01 08 0a 42 0f c2 e2   ·····F·· ····B···
0040  9a c6 45 e1 72 65 76 72  65 73 2f 4e 43 2f 73 64   ··E·revr es/NC/sd
0050  61 6f 6c 6e 77 6f 44 2f  79 6f 6a 2f 65 6d 6f 68   aolnwoD/ yoj/emoh
0060  2f 20                                              /
```

Server sent the data in reversed format

Ls:

No.	Time	Source	Destination	Protocol	Length	Info
37	79.398502586	10.0.2.15	35.232.111.17	TCP	56	47416 → 80 [FIN, ACK] Seq=88 Ack=150 Win=64091 Len=0
38	79.399006553	35.232.111.17	10.0.2.15	TCP	62	80 → 47416 [ACK] Seq=150 Ack=89 Win=65535 Len=0
39	80.407196480	35.232.111.17	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
40	80.407322016	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0
41	80.407568266	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [FIN, ACK] Seq=88 Ack=149 Win=64092 Len=0
42	80.408161969	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [ACK] Seq=149 Ack=89 Win=65535 Len=0
43	80.420175420	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [FIN, ACK] Seq=149 Ack=89 Win=65535 Len=0
44	80.420217039	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=89 Ack=150 Win=64092 Len=0
51	90.546689696	127.0.0.1	127.0.1.1	TCP	71	52888 → 4040 [PSH, ACK] Seq=5 Ack=31 Win=512 Len=3 TSval=2596…
52	90.546723222	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=31 Ack=8 Win=512 Len=0 TSval=110841973…
53	90.547026008	127.0.1.1	127.0.0.1	TCP	157	4040 → 52888 [PSH, ACK] Seq=31 Ack=8 Win=512 Len=89 TSval=110…
54	90.547036017	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=8 Ack=120 Win=512 Len=0 TSval=25967758…

```
▸ Frame 51: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52888, Dst Port: 4040, Seq: 5, Ack: 31, Len: 3
▾ Data (3 bytes)
    Data: 736c20
    [Length: 3]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 37 85 f7 40 00  40 06 b5 c7 7f 00 00 01   E··7··@· @·......
0020  7f 00 01 01 ce 98 0f c8  87 34 2c d7 7c 51 ba 96   ........ ·4,·|Q··
0030  80 18 02 00 ff 2b 00 00  01 01 08 0a 9a c7 a7 94   ·····+·· ........
0040  42 0f c2 e2 73 6c 20                               B···sl
```

client sent the message in encrypted format which means it reversed “ls” to “sl”

No.	Time	Source	Destination	Protocol	Length	Info
37	79.398502586	10.0.2.15	35.232.111.17	TCP	56	47416 → 80 [FIN, ACK] Seq=88 Ack=150 Win=64091 Len=0
38	79.399006553	35.232.111.17	10.0.2.15	TCP	62	80 → 47416 [ACK] Seq=150 Ack=89 Win=65535 Len=0
39	80.407196480	35.232.111.17	10.0.2.15	HTTP	204	HTTP/1.1 204 No Content
40	80.407322016	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=88 Ack=149 Win=64092 Len=0
41	80.407568266	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [FIN, ACK] Seq=88 Ack=149 Win=64092 Len=0
42	80.408161969	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [ACK] Seq=149 Ack=89 Win=65535 Len=0
43	80.420175420	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [FIN, ACK] Seq=149 Ack=89 Win=65535 Len=0
44	80.420217039	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=89 Ack=150 Win=64092 Len=0
51	90.546689696	127.0.0.1	127.0.1.1	TCP	71	52888 → 4040 [PSH, ACK] Seq=5 Ack=31 Win=512 Len=3 TSval=2596…
52	90.546723222	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=31 Ack=8 Win=512 Len=0 TSval=110841973…
53	90.547026008	127.0.1.1	127.0.0.1	TCP	157	4040 → 52888 [PSH, ACK] Seq=31 Ack=8 Win=512 Len=89 TSval=110…
54	90.547036017	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=8 Ack=120 Win=512 Len=0 TSval=25967758…

```
▸ Frame 53: 157 bytes on wire (1256 bits), 157 bytes captured (1256 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52888, Seq: 31, Ack: 8, Len: 89
▾ Data (89 bytes)
    Data: 2c277478742e747874275b202c2779702e6e6f6974636e75…
    [Length: 89]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 04 06 08 00   ........ ........
0010  45 00 00 8d 89 74 40 00  40 06 b1 f4 7f 00 01 01   E····t@· @·......
0020  7f 00 00 01 0f c8 ce 98  7c 51 ba 96 87 34 2c da   ........ |Q···4,·
0030  80 18 02 00 ff 81 00 00  01 01 08 0a 42 11 24 95   ........ ····B·$·
0040  9a c7 a7 94 2c 27 74 78  74 2e 74 78 74 27 5b 20   ····,'tx t.txt'[ 
0050  2c 27 79 70 2e 6e 6f 69  74 63 6e 75 66 27 20 2c   ,'yp.noi tcnuf' ,
0060  27 31 66 27 20 2c 27 79  70 2e 72 65 76 72 65 73   '1f' ,'y p.revres
0070  27 20 2c 27 74 78 74 2e  32 74 78 74 27 20 2c 27   ' ,'txt. 2txt' ,'
0080  67 6e 70 2e 63 65 74 69  68 63 72 61 27 20 5d 27   gnp.ceti hcra' ]'
```

Server sent the data in reversed format

cd:

No.	Time	Source	Destination	Protocol	Length	Info
41	80.407568266	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [FIN, ACK] Seq=88 Ack=149 Win=64092 Len=0
42	80.408161969	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [ACK] Seq=149 Ack=89 Win=65535 Len=0
43	80.420175420	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [FIN, ACK] Seq=149 Ack=89 Win=65535 Len=0
44	80.420217039	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=89 Ack=150 Win=64092 Len=0
51	90.546689696	127.0.0.1	127.0.1.1	TCP	71	52888 → 4040 [PSH, ACK] Seq=5 Ack=31 Win=512 Len=3 TSval=2596…
52	90.546723222	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=31 Ack=8 Win=512 Len=0 TSval=110841973…
53	90.547026008	127.0.1.1	127.0.0.1	TCP	157	4040 → 52888 [PSH, ACK] Seq=31 Ack=8 Win=512 Len=89 TSval=110…
54	90.547036017	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=8 Ack=120 Win=512 Len=0 TSval=25967758…
61	173.041040914	127.0.0.1	127.0.1.1	TCP	74	52888 → 4040 [PSH, ACK] Seq=8 Ack=120 Win=512 Len=6 TSval=259…
62	173.041106065	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=120 Ack=14 Win=512 Len=0 TSval=1108502…
63	173.041331823	127.0.1.1	127.0.0.1	TCP	101	4040 → 52888 [PSH, ACK] Seq=120 Ack=14 Win=512 Len=33 TSval=1…
64	173.041342740	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=14 Ack=153 Win=512 Len=0 TSval=2596858…

```
▸ Frame 61: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 52888, Dst Port: 4040, Seq: 8, Ack: 120, Len: 6
▾ Data (6 bytes)
    Data: 646320316620
    [Length: 6]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 01 08 00   ........ ........
0010  45 00 00 3a 85 f9 40 00  40 06 b5 c2 7f 00 00 01   E··:··@· @·......
0020  7f 00 01 01 ce 98 0f c8  87 34 2c da 7c 51 ba ef   ........ ·4,·|Q··
0030  80 18 02 00 ff 2e 00 00  01 01 08 0a 9a c8 e9 d2   ........ ........
0040  42 11 24 95 64 63 20 31  66 20                     B·$·dc 1 f
```

Client reversed the command in word by word manner due to which "cd f1" becomes "dc 1f"

No.	Time	Source	Destination	Protocol	Length	Info
41	80.407568266	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [FIN, ACK] Seq=88 Ack=149 Win=64092 Len=0
42	80.408161969	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [ACK] Seq=149 Ack=89 Win=65535 Len=0
43	80.420175420	35.232.111.17	10.0.2.15	TCP	62	80 → 47418 [FIN, ACK] Seq=149 Ack=89 Win=65535 Len=0
44	80.420217039	10.0.2.15	35.232.111.17	TCP	56	47418 → 80 [ACK] Seq=89 Ack=150 Win=64092 Len=0
51	90.546689696	127.0.0.1	127.0.1.1	TCP	71	52888 → 4040 [PSH, ACK] Seq=5 Ack=31 Win=512 Len=3 TSval=2596…
52	90.546723222	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=31 Ack=8 Win=512 Len=0 TSval=110841973…
53	90.547026008	127.0.1.1	127.0.0.1	TCP	157	4040 → 52888 [PSH, ACK] Seq=31 Ack=8 Win=512 Len=89 TSval=110…
54	90.547036017	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=8 Ack=120 Win=512 Len=0 TSval=25967758…
61	173.041040914	127.0.0.1	127.0.1.1	TCP	74	52888 → 4040 [PSH, ACK] Seq=8 Ack=120 Win=512 Len=6 TSval=259…
62	173.041106065	127.0.1.1	127.0.0.1	TCP	68	4040 → 52888 [ACK] Seq=120 Ack=14 Win=512 Len=0 TSval=1108502…
63	173.041331823	127.0.1.1	127.0.0.1	TCP	101	4040 → 52888 [PSH, ACK] Seq=120 Ack=14 Win=512 Len=33 TSval=1…
64	173.041342740	127.0.0.1	127.0.1.1	TCP	68	52888 → 4040 [ACK] Seq=14 Ack=153 Win=512 Len=0 TSval=2596858…

```
▸ Frame 63: 101 bytes on wire (808 bits), 101 bytes captured (808 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52888, Seq: 120, Ack: 14, Len: 33
▾ Data (33 bytes)
    Data: 79726f746365726944207361682066e656562206465676e61…
    [Length: 33]
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 55 89 76 40 00  40 06 b2 2a 7f 00 01 01   E··U·v@· @··*····
0020  7f 00 00 01 0f c8 ce 98  7c 51 ba ef 87 34 2c e0   ........ |Q···4,·
0030  80 18 02 00 ff 49 00 00  01 01 08 0a 42 12 66 d3   ·····I·· ····B·f·
0040  9a c8 e9 d2 79 72 6f 74  63 65 72 69 44 20 73 61   ····yrot ceriD sa
0050  68 20 6e 65 65 62 20 64  65 67 6e 61 68 63 20 6f   h neeb d egnahc o
0060  74 20 31 66 20                                      t 1f
```

Server also sent the message in word by word reverse manner

DWD and UPD:

No.	Time	Source	Destination	Protocol	Lengt	Info
2	0.000030497	127.0.1.1	127.0.0.1	TCP	68	4040 → 52922 [ACK] Seq=1 Ack=10 Win=512 Len=0 TSval=111325802…
4	6.561287776	127.0.1.1	127.0.0.1	TCP	68	4040 → 52922 [ACK] Seq=1 Ack=17 Win=512 Len=0 TSval=111326458…
6	6.562514337	127.0.0.1	127.0.1.1	TCP	68	52922 → 4040 [ACK] Seq=17 Ack=14 Win=512 Len=0 TSval=26016206…
8	6.562598663	127.0.0.1	127.0.1.1	TCP	68	52922 → 4040 [ACK] Seq=17 Ack=25 Win=512 Len=0 TSval=26016206…
9	6.562621284	127.0.1.1	127.0.0.1	TCP	68	4040 → 52922 [FIN, ACK] Seq=25 Ack=17 Win=512 Len=0 TSval=111…
10	6.563054299	127.0.0.1	127.0.1.1	TCP	68	52922 → 4040 [FIN, ACK] Seq=17 Ack=26 Win=512 Len=0 TSval=260…
11	6.563189979	127.0.1.1	127.0.0.1	TCP	68	4040 → 52922 [ACK] Seq=26 Ack=18 Win=512 Len=0 TSval=11132645…
3	6.561272397	127.0.0.1	127.0.1.1	TCP	75	52922 → 4040 [PSH, ACK] Seq=10 Ack=1 Win=512 Len=7 TSval=2601…
1	0.000000000	127.0.0.1	127.0.1.1	TCP	77	52922 → 4040 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=9 TSval=26016…
7	6.562592529	127.0.1.1	127.0.0.1	TCP	79	4040 → 52922 [PSH, ACK] Seq=14 Ack=17 Win=512 Len=11 TSval=11…
5	6.562501354	127.0.1.1	127.0.0.1	TCP	81	4040 → 52922 [PSH, ACK] Seq=1 Ack=17 Win=512 Len=13 TSval=111…

```
▸ Frame 5: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4040, Dst Port: 52922, Seq: 1, Ack: 17, Len: 13
▸ Data (13 bytes)
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ····· ····· ·····
0010  45 00 00 41 6d b6 40 00  40 06 cd fe 7f 00 01 01   E··Am·@· @·····
0020  7f 00 00 01 0f c8 ce ba  20 ee 7e 33 c0 ed e6 af   ····· ··~3····
0030  80 18 02 00 ff 35 00 00  01 01 08 0a 42 5b 11 cb   ·····5·· ····B[··
0040  9b 11 94 c9 6f 6c 6c 65  48 20 21 64 6c 72 6f 57   ····olle H !dlroW
0050  20
```

server reversed the data in word by word manner due to which “Hello world!” becomes “olleH !dlrow” and client will write this data after decryption. In upload, client will encrypt and send while server will receive and decrypt the data