

Secure File Transfer Application

...

Asymmetric Encryption with SSL Certificates

Group Details

Group Number: 6

Group Members

- | | | |
|----|---------------------|----------|
| 1. | Ajay Dayma | 17114006 |
| 2. | Bhavye Jain | 17114020 |
| 3. | Kaustubh Trivedi | 17114044 |
| 4. | Shiva Reddy | 17114045 |
| 5. | Sai Krishna Abhiram | 17114054 |
| 6. | Ritik Kumar | 17114063 |
| 7. | Saurabh Singh | 17114068 |

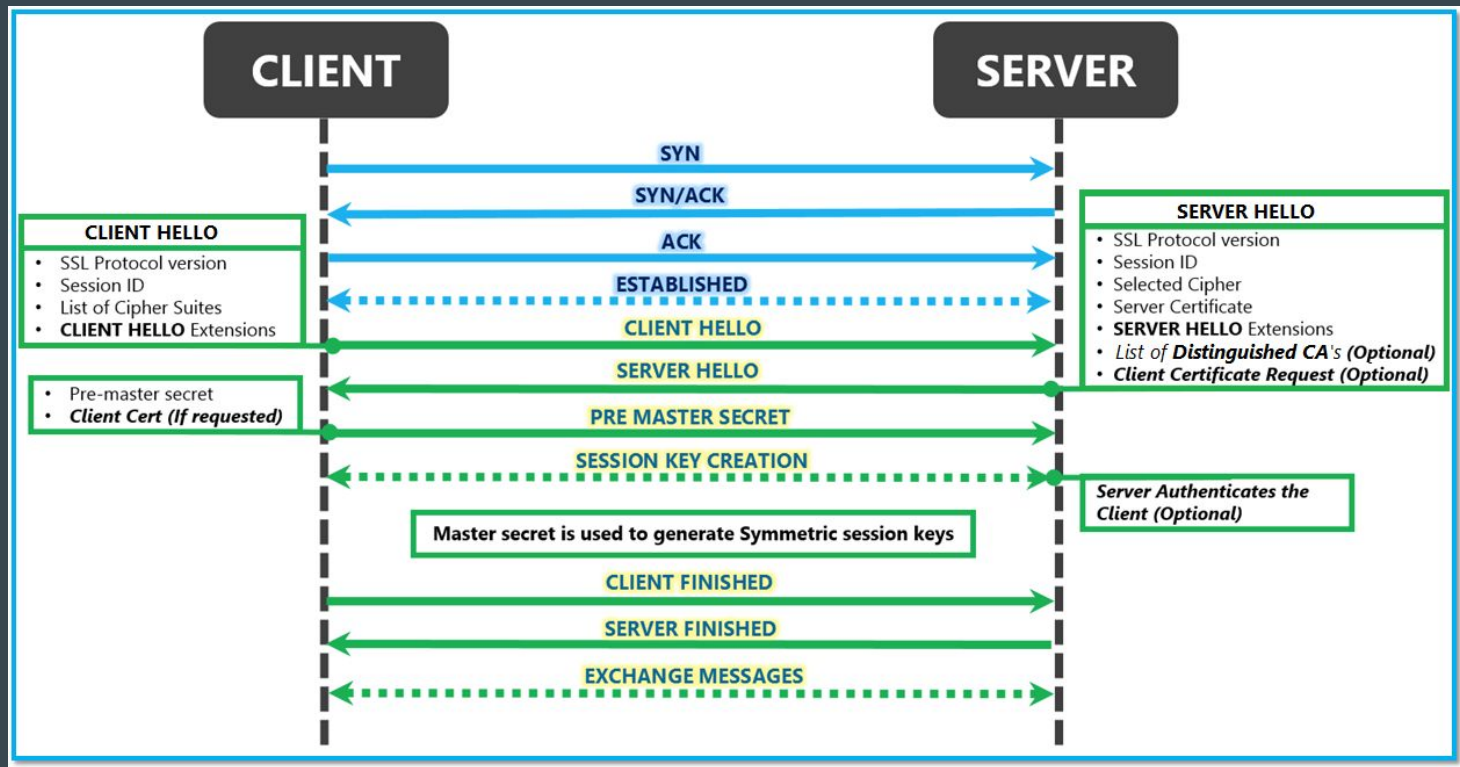
The Problem Statement

An organization needs an application which can help their employees to transfer files between them securely on the same network. Develop an application using socket programming to send files between two machines and secure the data transfer using a strong encryption algorithm. Capture these packets using a sniffing tool like Wireshark and show that data transfer is secure

Assumptions

1. The required system is built to operate inside the network of an organization.
2. Since the clients belong to the same organization, they need not communicate to decide a cypher between them.
3. The certificates for secure communication are digitally generated by a master authentication server inside the organization. This server has the root certificate and key pair.
4. All the clients agree to encrypt data using asymmetric public-private key encryption.
5. The public ports of other nodes are already known within the network.
6. Every device entering our network already has the organization's root certificate pre-installed on their devices, which would be used for key chain validation.
7. Auth server is online and it's accessible to the user in the network.

Security on the internet - TLS/SSL



Security in our application

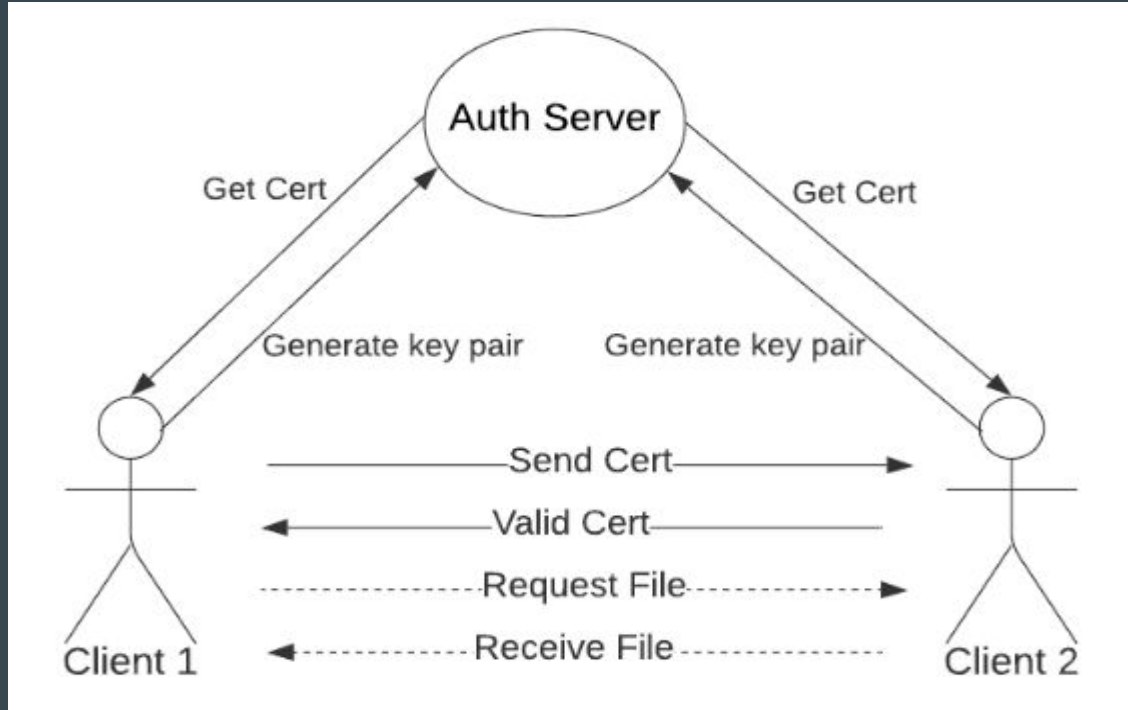
Both the sender and receiver have a certificate issued by the authentication server. First, a TCP handshake is done between the two parties while creating a socket.

Further steps-

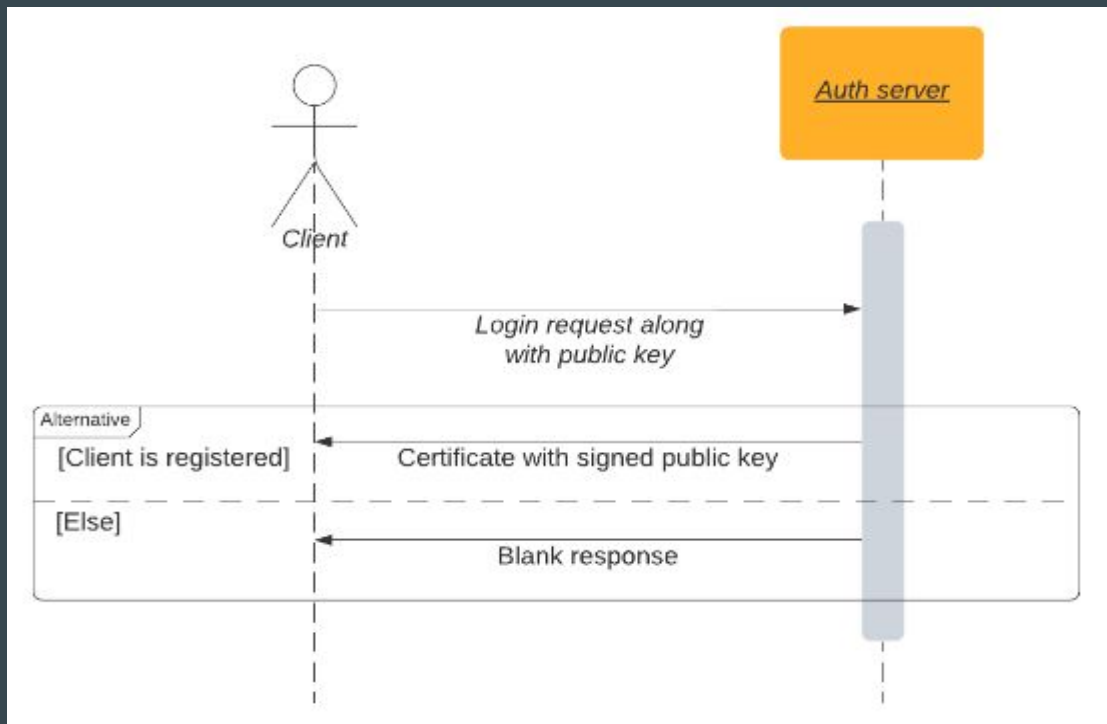
- Client 1 Certificate Send: The client that wants to initiate a file transfer sends a request for a secure connection with the server. It sends its signed certificate to the other client.
- Client 2 Certificate Auth and Send: The receiver receives the transfer request, and verifies the authenticity of the certificate sent by client 1 using the auth server's trust anchor certificate. The secure connection is then established.

Working of the System

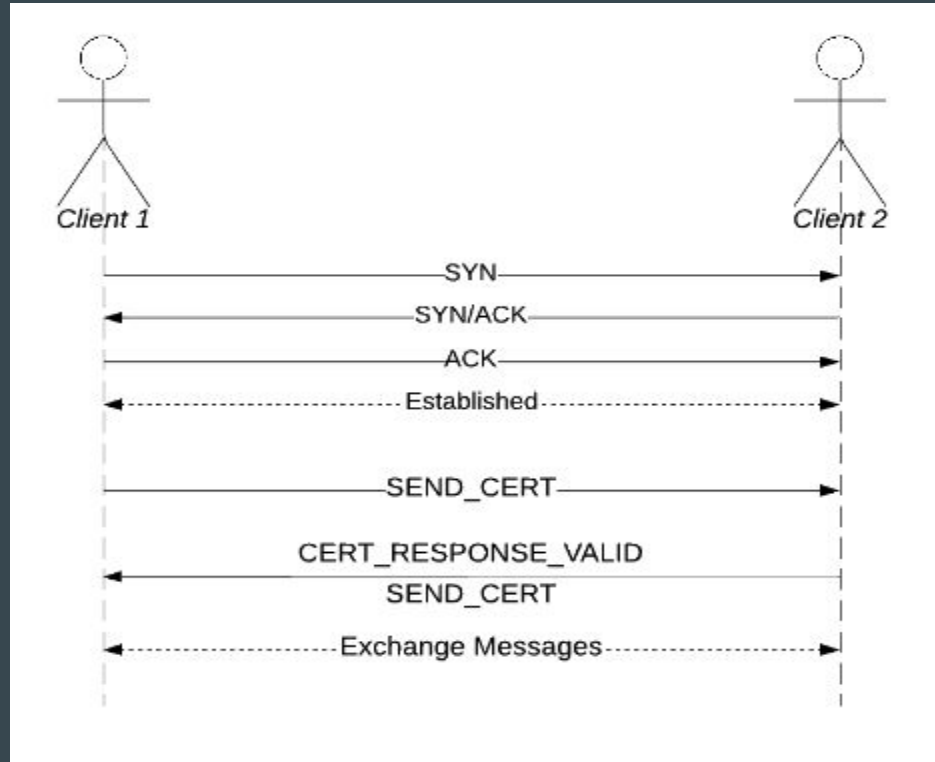
Architecture



Client Auth Server



Client to client communication



Code Components [Modularization]

- auth.py - The master authentication server for the organization. Functions as the trust anchor for the system and signs certificates of the clients.
 - SHA256 hashing to encrypt the certificate
 - X.509 format for the certificate
- client.py - The front end script for each individual client.
 - Gets the user logged in safely and gets the user's key certified.
 - Initiates a connection to another client for file transfer

Code Components [Modularization]

- peer.py - File to store the state of a client
 - Stored data like received public key for a particular connection.
 - Also responsible for sending and receiving data from connected peer.
 - Data sent is in the form of a struct: `!4sL<payload_length>s`
- connection.py - Helper script containing procedures for requesting files and to handle incoming connections and messages
- encrypt.py - A helper script to handle and generate keys and certificates

Working Demonstration

```
(venv) codebase (master*) » py auth.py
Before starting authserver, you can add new users here...
Add new user? (t/f) t
Enter Alias: user3
Enter Password: password3
Add new user? (t/f) f
Auth server running.
127.0.0.1 - - [19/Nov/2020 00:15:51] "POST /verify HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2020 00:16:03] "POST /verify HTTP/1.1" 200 -
```

Working Demonstration

```
(venv) codebase (master*) » py client.py ~/Projects/random/new/ritik/Se
Please enter your corp alias: user1
Please enter your password: password1
Enter port to listen on: 1990
Listening for incoming connections on port localhost:1990
[Thread-1] Server started: (localhost:1990)
Do you like to request a file? t/f? t
Enter peer's port: 1991
Enter the file name: file1_.txt
Requesting for file file1_.txt on localhost:1991
[MainThread] Sent SECE
[MainThread] Received certificate
[MainThread] Sent RQFL
[MainThread] File received written to: received_file1_.txt
Do you like to request a file? t/f? [Thread-2] New child Thread-2
[Thread-2] Connected ('127.0.0.1', 59594)
[Thread-2] Handling peer msg
[Thread-2] Handling peer msg
File request for: b'file1_.png'
[Thread-2] Disconnecting ('127.0.0.1', 59594)
```

Working Demonstration

```
(venv) codebase (master*) » py client.py      ~/Projects/random/new/ritik/Se
Please enter your corp alias: user3
Please enter your password: password3
Enter port to listen on: 1991
Listening for incoming connections on port localhost:1991
[Thread-1] Server started: (localhost:1991)
Do you like to request a file? t/f? [Thread-2] New child Thread-2
[Thread-2] Connected ('127.0.0.1', 56686)
[Thread-2] Handling peer msg
[Thread-2] Handling peer msg
File request for: b'file1_.txt'
[Thread-2] Disconnecting ('127.0.0.1', 56686)
t
Enter peer's port: 1990
Enter the file name: file1_.png
Requesting for file file1_.png on localhost:1990
[MainThread] Sent SECE
[MainThread] Received certificate
[MainThread] Sent RQFL
[MainThread] File received written to: received_file1_.png
Do you like to request a file? t/f? ☐
```

Working Demonstration

[illegible]

Wireshark Captures

DATA EXCHANGES

not dns					
No.	Time	Source	Destination	Protocol	Length Info
9	38.664211768	127.0.0.1	127.0.0.1	TCP	74 55990 → 12565 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=
10	38.664236453	127.0.0.1	127.0.0.1	TCP	74 12565 → 55990 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=6549
11	38.664249683	127.0.0.1	127.0.0.1	TCP	66 55990 → 12565 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3639644
12	38.664285978	127.0.0.1	127.0.0.1	TCP	241 55990 → 12565 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=175 TSval=
13	38.664290683	127.0.0.1	127.0.0.1	TCP	66 12565 → 55990 [ACK] Seq=1 Ack=176 Win=65408 Len=0 TSval=36396
14	38.664305898	127.0.0.1	127.0.0.1	HTTP	1217 POST /verify HTTP/1.1
15	38.664309486	127.0.0.1	127.0.0.1	TCP	66 12565 → 55990 [ACK] Seq=1 Ack=1327 Win=64384 Len=0 TSval=3639
16	38.690805293	127.0.0.1	127.0.0.1	TCP	198 12565 → 55990 [PSH, ACK] Seq=1 Ack=1327 Win=65536 Len=132 Tsv
17	38.690822521	127.0.0.1	127.0.0.1	TCP	66 55990 → 12565 [ACK] Seq=1327 Ack=133 Win=65408 Len=0 TSval=36
18	38.690854938	127.0.0.1	127.0.0.1	TCP	1620 12565 → 55990 [PSH, ACK] Seq=133 Ack=1327 Win=65536 Len=1554
19	38.690859732	127.0.0.1	127.0.0.1	TCP	66 55990 → 12565 [ACK] Seq=1327 Ack=1687 Win=64128 Len=0 TSval=3
20	38.690908130	127.0.0.1	127.0.0.1	HTTP	66 HTTP/1.0 200 OK (text/plain)
21	38.691367826	127.0.0.1	127.0.0.1	TCP	66 55990 → 12565 [FIN, ACK] Seq=1327 Ack=1688 Win=65536 Len=0 TS
22	38.691377339	127.0.0.1	127.0.0.1	TCP	66 12565 → 55990 [ACK] Seq=1688 Ack=1328 Win=65536 Len=0 TSval=3
23	59.201028927	127.0.0.1	127.0.0.1	TCP	74 55992 → 12565 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=
24	59.201042646	127.0.0.1	127.0.0.1	TCP	74 12565 → 55992 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=6549
25	59.201051171	127.0.0.1	127.0.0.1	TCP	66 55992 → 12565 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3639655
26	59.201076996	127.0.0.1	127.0.0.1	TCP	241 55992 → 12565 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=175 TSval=
27	59.201080517	127.0.0.1	127.0.0.1	TCP	66 12565 → 55992 [ACK] Seq=1 Ack=176 Win=65408 Len=0 TSval=36396
28	59.201090542	127.0.0.1	127.0.0.1	HTTP	1217 POST /verify HTTP/1.1
29	59.201093125	127.0.0.1	127.0.0.1	TCP	66 12565 → 55992 [ACK] Seq=1 Ack=1327 Win=64384 Len=0 TSval=3639
30	59.262868594	127.0.0.1	127.0.0.1	TCP	198 12565 → 55992 [PSH, ACK] Seq=1 Ack=1327 Win=65536 Len=132 Tsv
31	59.262884066	127.0.0.1	127.0.0.1	TCP	66 55992 → 12565 [ACK] Seq=1327 Ack=133 Win=65408 Len=0 TSval=36
32	59.263555808	127.0.0.1	127.0.0.1	TCP	1620 12565 → 55992 [PSH, ACK] Seq=133 Ack=1327 Win=65536 Len=1554
33	59.263565130	127.0.0.1	127.0.0.1	TCP	66 55992 → 12565 [ACK] Seq=1327 Ack=1687 Win=64128 Len=0 TSval=3
34	59.263612037	127.0.0.1	127.0.0.1	HTTP	66 HTTP/1.0 200 OK (text/plain)
35	59.263608672	127.0.0.1	127.0.0.1	TCP	66 55992 → 12565 [FIN, ACK] Seq=1327 Ack=1688 Win=65536 Len=0 Tsv
36	59.263688332	127.0.0.1	127.0.0.1	TCP	66 12565 → 55992 [ACK] Seq=1688 Ack=1328 Win=65536 Len=0 TSval=3
49	131.347572855	127.0.0.1	127.0.0.1	TCP	74 56686 → 1991 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=
50	131.347606118	127.0.0.1	127.0.0.1	TCP	74 1991 → 56686 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495
51	131.347626451	127.0.0.1	127.0.0.1	TCP	66 56686 → 1991 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=36397370
52	131.348292296	127.0.0.1	127.0.0.1	TCP	1628 56686 → 1991 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1562 TSval=
53	131.348311623	127.0.0.1	127.0.0.1	TCP	66 1991 → 56686 [ACK] Seq=1 Ack=1563 Win=64128 Len=0 TSval=36397
54	131.350179195	127.0.0.1	127.0.0.1	TCP	1628 1991 → 56686 [PSH, ACK] Seq=1 Ack=1563 Win=65536 Len=1562 Tsv
55	131.350205785	127.0.0.1	127.0.0.1	TCP	66 56686 → 1991 [ACK] Seq=1563 Ack=1563 Win=64128 Len=0 TSval=36
56	131.352932974	127.0.0.1	127.0.0.1	TCP	330 56686 → 1991 [PSH, ACK] Seq=1563 Ack=1563 Win=65536 Len=264 T
57	131.352973790	127.0.0.1	127.0.0.1	TCP	66 1991 → 56686 [ACK] Seq=1563 Ack=1827 Win=65280 Len=0 TSval=36
58	131.359756895	127.0.0.1	127.0.0.1	TCP	330 1991 → 56686 [PSH, ACK] Seq=1563 Ack=1827 Win=65536 Len=264 T
59	131.359793113	127.0.0.1	127.0.0.1	TCP	66 56686 → 1991 [ACK] Seq=1827 Ack=1827 Win=65280 Len=0 TSval=36
60	131.360011036	127.0.0.1	127.0.0.1	TCP	66 1991 → 56686 [FIN, ACK] Seq=1827 Ack=1827 Win=65536 Len=0 Tsv
61	131.365946904	127.0.0.1	127.0.0.1	TCP	66 56686 → 1991 [FIN, ACK] Seq=1827 Ack=1828 Win=65536 Len=0 Tsv
62	131.365982714	127.0.0.1	127.0.0.1	TCP	66 1991 → 56686 [ACK] Seq=1828 Ack=1828 Win=65536 Len=0 TSval=36
327	264.589958672	127.0.0.1	127.0.0.1	TCP	74 55994 → 1990 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=
328	264.589995572	127.0.0.1	127.0.0.1	TCP	74 1990 → 55994 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495
329	264.590034845	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=36398703
330	264.590090246	127.0.0.1	127.0.0.1	TCP	1628 55994 → 1990 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1562 TSval=
331	264.591017420	127.0.0.1	127.0.0.1	TCP	66 1990 → 55994 [ACK] Seq=1 Ack=1563 Win=64128 Len=0 TSval=36398
332	264.595769985	127.0.0.1	127.0.0.1	TCP	1628 1990 → 55994 [PSH, ACK] Seq=1 Ack=1563 Win=65536 Len=1562 Tsv
333	264.595810190	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [ACK] Seq=1563 Ack=1563 Win=64128 Len=0 TSval=36
334	264.598208664	127.0.0.1	127.0.0.1	TCP	330 55994 → 1990 [PSH, ACK] Seq=1563 Ack=1563 Win=65536 Len=264 T
335	264.598967731	127.0.0.1	127.0.0.1	TCP	66 1990 → 55994 [ACK] Seq=1563 Ack=1827 Win=65280 Len=0 TSval=36
336	264.605043835	127.0.0.1	127.0.0.1	TCP	330 1990 → 55994 [PSH, ACK] Seq=1563 Ack=1827 Win=65536 Len=264 T
337	264.605078270	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [ACK] Seq=1827 Ack=1827 Win=65280 Len=0 TSval=36
338	264.605402408	127.0.0.1	127.0.0.1	TCP	330 1990 → 55994 [PSH, ACK] Seq=1827 Ack=1827 Win=65536 Len=264 T
339	264.605419917	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [ACK] Seq=1827 Ack=2091 Win=65824 Len=0 TSval=36
340	264.605742955	127.0.0.1	127.0.0.1	TCP	330 1990 → 55994 [PSH, ACK] Seq=2091 Ack=1827 Win=65536 Len=264 T
341	264.605756750	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [ACK] Seq=1827 Ack=2355 Win=64768 Len=0 TSval=36
342	264.605922226	127.0.0.1	127.0.0.1	TCP	66 1990 → 55994 [FIN, ACK] Seq=2355 Ack=1827 Win=65536 Len=0 Tsv
343	264.627517575	127.0.0.1	127.0.0.1	TCP	66 55994 → 1990 [FIN, ACK] Seq=1827 Ack=2356 Win=65536 Len=0 Tsv
344	264.627561873	127.0.0.1	127.0.0.1	TCP	66 1990 → 55994 [ACK] Seq=2356 Ack=1828 Win=65536 Len=0 TSval=36

No.	Time	Source	Destination	Protocol	Length	Info
31	50.262884866	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [ACK] Seq=1327 Ack=133 Win=65408 Len=0 TSval=36..
32	50.263555808	127.0.0.1	127.0.0.1	TCP	1628	12565 → 55992 [PSH, ACK] Seq=133 Ack=1327 Win=65536 Len=1554 ..
33	50.263665138	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [ACK] Seq=1327 Ack=1687 Win=64128 Len=0 TSval=3..
34	50.263612037	127.0.0.1	127.0.0.1	HTTP	66	HTTP/1.0 200 OK (text/plain)
35	50.263688672	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [FIN, ACK] Seq=1327 Ack=1688 Win=65536 Len=0 TS..
36	50.263688332	127.0.0.1	127.0.0.1	TCP	66	12565 → 55992 [ACK] Seq=1688 Ack=1328 Win=65536 Len=0 TSval=3..
37	50.263787205	127.0.0.1	127.0.0.1	TCP	74	56686 → 1991 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=..
50	131.347606118	127.0.0.1	127.0.0.1	TCP	74	1991 → 56686 [SYN, ACK] Seq=1 Ack=1 Win=65483 Len=0 MSS=65495..
51	131.347626451	127.0.0.1	127.0.0.1	TCP	66	56686 → 1991 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=36397370..
52	131.347626225	127.0.0.1	127.0.0.1	TCP	1628	56686 → 1991 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=36397..
53	131.348311623	127.0.0.1	127.0.0.1	TCP	66	1991 → 56686 [ACK] Seq=1 Ack=1563 Win=64128 Len=0 TSval=36397..
54	131.350179195	127.0.0.1	127.0.0.1	TCP	1628	1991 → 56686 [PSH, ACK] Seq=1 Ack=1563 Win=65536 Len=1562 TSV..
55	131.350285785	127.0.0.1	127.0.0.1	TCP	66	56686 → 1991 [ACK] Seq=1563 Ack=1563 Win=64128 Len=0 TSval=36..
56	131.352923074	127.0.0.1	127.0.0.1	TCP	330	56686 → 1991 [PSH, ACK] Seq=1563 Ack=1563 Win=65536 Len=264 T..
57	131.352973790	127.0.0.1	127.0.0.1	TCP	66	1991 → 56686 [ACK] Seq=1563 Ack=1827 Win=65280 Len=0 TSval=36..
58	131.359756685	127.0.0.1	127.0.0.1	TCP	330	1991 → 56686 [PSH, ACK] Seq=1563 Ack=1827 Win=65536 Len=264 T..
59	131.359793113	127.0.0.1	127.0.0.1	TCP	66	56686 → 1991 [ACK] Seq=1827 Ack=1827 Win=65280 Len=0 TSval=36..
Frame 52: 1628 bytes on wire (13024 bits), 1628 bytes captured (13024 bits) on interface lo, id 0						
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)						
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1						
Transmission Control Protocol, Src Port: 56686, Dst Port: 1991, Seq: 1, Ack: 1, Len: 1562						
0000	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	45 00	E.....E..		
0010	06 4e f8 00 40 00 40 06	3c 9c ff 00 00 00 01 7f	00	N...>.....		
0020	00 01 dd 06 e7 c7 42 b9	71 0a 66 f0 cf 81 90 18		...n...q f....		
0030	02 00 04 43 00 00 01 01	08 0a d8 f1 fb 09 d8 f1		...C.....		
0040	fb 09 53 45 43 43 45 00	06 12 2d 2d 2d 2d 2d 42	00	...SECE-----B		
0050	45 47 49 4e 20 43 45 52	54 49 46 49 43 41 54 45	00	EGIN CR TIF TICATE		
0060	2d 2d 2d 2d 2d 2d 2d 49	49 45 54 64 43 41 6a	00	...MI IETJCA(....		
0070	61 67 41 77 49 42 41 67	49 55 45 4e 71 67 79 32	00	aQaaIbaQ IUeqy2		
0080	72 30 61 61 59 51 2f 58	46 35 46 4d 66 7a 2f 51	00	r0aaYQ/X FPFmfzF		
0090	59 44 43 51 45 77 44 51	59 4a 4b 6f 5a 49 68 76	00	YDCeqWQ YJKozThv		
0100	43 41 51 51 51 4c 6a 42	51 41 77 64 54 45 4c 4d	00	ciAQEL B QAwitTELm		
0110	41 6b 47 41 31 55 45 42	68 4d 43 53 55 54 3a 78	43	AkGA1UEB HMCsu4X		
0120	7a 41 4a 42 67 41 56 42	41 67 40 41 61 56 4d 40	00	ZAJBgNBv AgMALIVL		
0130	51 73 77 43 51 59 44 56	51 51 48 44 41 4a 53 53	00	QswCQVdV QKHAJSS		
0140	7a 4e 4d 41 73 47 0a 41	41 31 55 43 67 77 45	00	ZENMASG ALUEGwG		
0150	58 65 66 55 55 6a 45 4d	4d 41 67 47 41 31 55 45	00	SUIUjUen MAoGAUe		

Encrypted data

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

not dns

No.	Time	Source	Destination	Protocol	Length	Info
31	50.262884066	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [ACK] Seq=1327 Ack=133 Win=65408 Len=0 TSval=36...
32	50.263555808	127.0.0.1	127.0.0.1	TCP	1620	12565 → 55992 [PSH, ACK] Seq=133 Ack=1327 Win=65536 Len=1554 ...
33	50.263565130	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [ACK] Seq=1327 Ack=1687 Win=64128 Len=0 TSval=3...
34	50.263612937	127.0.0.1	127.0.0.1	HTTP	66	HTTP/1.0 200 OK (text/plain)
35	50.263680672	127.0.0.1	127.0.0.1	TCP	66	55992 → 12565 [FIN, ACK] Seq=1327 Ack=1688 Win=65536 Len=0 TS...
36	50.263688332	127.0.0.1	127.0.0.1	TCP	66	12565 → 55992 [ACK] Seq=1688 Ack=1328 Win=65536 Len=0 TSval=3...
49	131.347572055	127.0.0.1	127.0.0.1	TCP	74	56886 → 1991 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=...
50	131.347606118	127.0.0.1	127.0.0.1	TCP	74	1991 → 56886 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495...
51	131.347626451	127.0.0.1	127.0.0.1	TCP	66	56886 → 1991 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=36397370...
52	131.348292296	127.0.0.1	127.0.0.1	TCP	1628	56886 → 1991 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1562 TSval=...
53	131.348311623	127.0.0.1	127.0.0.1	TCP	66	1991 → 56886 [ACK] Seq=1 Ack=1563 Win=64128 Len=0 TSval=36397...
54	131.350179195	127.0.0.1	127.0.0.1	TCP	1628	1991 → 56886 [PSH, ACK] Seq=1 Ack=1563 Win=65536 Len=1562 TSv...
55	131.350205785	127.0.0.1	127.0.0.1	TCP	66	56886 → 1991 [ACK] Seq=1563 Ack=1563 Win=64128 Len=0 TSval=36...
56	131.352932974	127.0.0.1	127.0.0.1	TCP	330	56886 → 1991 [PSH, ACK] Seq=1563 Ack=1563 Win=65536 Len=264 T...
57	131.352973790	127.0.0.1	127.0.0.1	TCP	66	1991 → 56886 [ACK] Seq=1563 Ack=1827 Win=65280 Len=0 TSval=36...
58	131.359756895	127.0.0.1	127.0.0.1	TCP	330	1991 → 56886 [PSH, ACK] Seq=1563 Ack=1827 Win=65536 Len=264 T...
59	131.359793113	127.0.0.1	127.0.0.1	TCP	66	56886 → 1991 [ACK] Seq=1827 Ack=1827 Win=65280 Len=0 TSval=36...

► Frame 58: 330 bytes on wire (2640 bits), 330 bytes captured (2640 bits) on interface lo, id 0

► Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)

► Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1

► Transmission Control Protocol, Src Port: 1991, Dst Port: 56886, Seq: 1563, Ack: 1827, Len: 264

```
0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00  .....E-
0010  01 3c e1 48 40 00 40 06 5a 71 7f 00 00 01 7f 00  <H0@Zq
0020  00 01 07 c7 dd 6e 66 f0 d5 9b a2 b9 78 2c 80 18  ....nf...x,
0030  02 00 ff 30 00 00 01 01 08 0a d8 f1 fb 15 d8 f1  ....0.....
0040  fb 0e 52 53 46 4c 00 00 01 00 0e 09 06 60 6f b8  ..RSFL....o
0050  aa d3 0a b4 1a 2b 19 f5 4c d8 5e af 9a 30 63 a8  ...+...L^..0c
0060  e8 f4 c4 28 c4 23 1d f2 45 23 5a 8e f0 31 a8 14  ...(#...E#Z..1
0070  2c 18 22 d5 54 12 5e 9b 32 d9 97 5b 00 8f 11 5c  ...T.A.2.[...
0080  f2 0d ba 4e e3 ef ef b8 63 dc 55 b4 06 1f 9a 80  ...N....c.U...
0090  91 42 45 04 52 36 0b a1 84 11 de 07 ad 2d 63 8e  ..BE6.....c
00a0  b7 a9 1b b6 2a 29 f8 7e 24 b9 4f 3f 19 6e 57 0c  ...*)~$.0?nw
00b0  50 56 6e a8 6e ba e0 c7 f5 ec d8 27 2e ca 74 d1  ..PVn-n....t
00c0  ed 7d c2 ce db a4 68 71 3b 80 46 7c ee 23 ea 86  ...}....hq;F|.#
00d0  2a f2 79 41 e1 93 44 8e 7a 1c 27 9e 5b 81 0e 42  *.yA.D.z.[.B
00e0  f1 31 f9 47 1e b2 04 f3 5e bf 92 b4 1a d9 29 95  ..1G.....A...
00f0  fd 5a 61 4b b4 4e 3e 50 d5 bf 9b c1 8c 22 c4 83  ..ZaK.N-P...."
0100  62 9f 9c ec 52 be b3 f8 77 d6 a0 c8 c2 27 bd d6  ..b..R....w....
0110  f0 4c 16 9b 00 07 3d 22 84 bb eb cf b5 50 1c e6  ..L....="....[
0120  d5 44 73 db 27 07 62 a5 03 44 61 70 f3 20 cc 35  ..Ds..b..Dap..5
0130  64 ec e0 74 f8 bf fc 0a 03 fe 22 fd 97 14 17 62  ..d..t....".b
0140  3a 75 40 68 a2 1f 6e a0 c4 f5                      ..ugh-n-...
```

Contributions

1. Ritik Kumar - Contributed to the actual coding the implementation together with looking at the available libraries to **handle encryption and key management**. Contributed to the **terminal based demonstration** for the project.
2. Kaustubh Trivedi - Contributed in the actual code, specifically **connections.py and peer.py**, where I handled the encryption and sending of data as bytes chunks from file. Contributed to **make the tool interactive** and demonstrate its POC.
3. Bhavye Jain - Created the **overall design** of the system encompassing the **various entities, their roles and interactions**. Evaluated the various options of signing certificates and decided on the hierarchy suitable for the project.
4. Saurabh Singh - Worked on **securing file transfer** using Sockets and developing an **architecture that works similar to SSL/TLS** security on the internet to ensure that communication is encrypted.
5. Ajay Dayma - Worked on security issues of File transfer and how we can improve the existing solution. **Explored different ways of securing communication**, specifically JWT.
6. Sai Krishna Abhiram - Packet Capture and Analysis using Wireshark
7. Shiva Reddy - Worked on what **assumptions to take for our System**. Explored different ways of securing communication, specifically **Symmetric Cryptography**.

Thank You