End-to-End Encrypted Messaging System with mailbox mechanism

Ishaan Agrawal

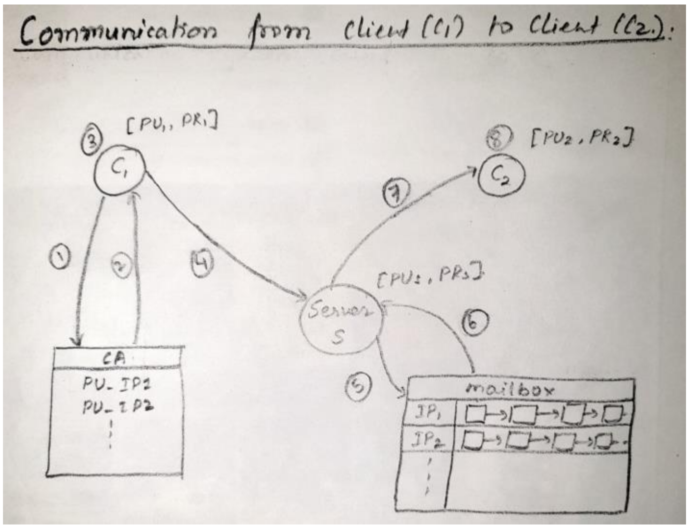
I. Introduction:

The problem addressed in the project is the need for secure communication between multiple clients over a network ensuring confidentiality of messages, as well as scalability and efficiency in handling concurrent clients.
The problem is imperative as it safeguards sensitive information from interception and compromise, ensuring privacy and security in contexts of personal communication.
Existing solutions, typically utilize Signal Protocols, Transport Layer Security (TLS), AES algorithm to achieve 'Confidentiality' and the server ensures message routing and message management by acting as a relay.
Key contributions include usage of OpenSSL library to implement RSA key generation and end-toend encryption and decryption for achieving confidentiality, a mailbox mechanism for efficient clientserver communication, and overall, ensuring secure and scalable messaging.

II. Objectives:

To implement end-to-end encryption using RSA algorithm to ensure confidentiality of messages transmitted between clients and the server.
To develop a scalable and efficient mailbox mechanism at the server to manage communication between multiple concurrent clients, minimizing delays and ensuring smooth operation.

III. Implementation and Results analysis:



Assumptions:

a) Public Keys of all the clients are stored by Certificate authority.

b) Each client has the IP address of all other clients in the network.

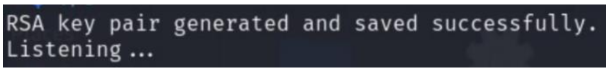
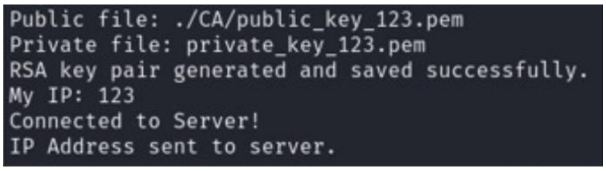
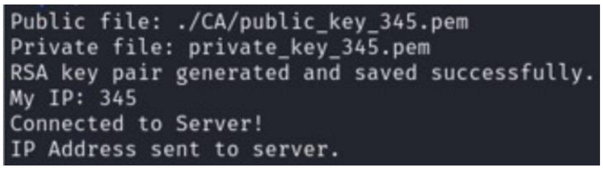
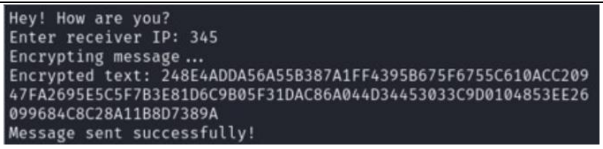
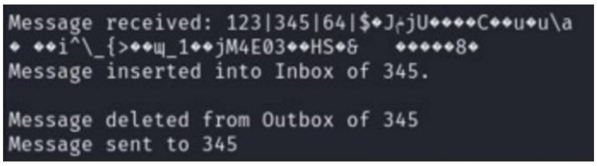
c) Communication is taking place between active clients, the ones which are connected to server.

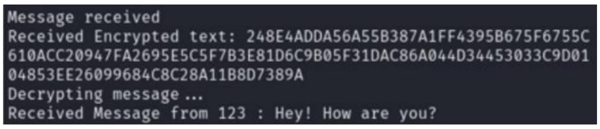
Steps Explanation:

1. The client that wishes to communicate(C1) generates its private and public keys using RSA Algorithm. The public keys are stored by Certificate Authority and it requests destination client's (C2) public key.
2. The client receives the desired client's public key from the Certificate Authority.
3. The client process performs encryption of message using the public key received.

4. The encrypted message of the form *‘sender|receiver|length|message’* is sent to the server for forwarding.
5. The server generates sender’s IP, receiver’s IP and the original message from the message received and stores the entire encrypted message destined for that receiver IP in the form of linked list.
6. For every entry in the mailbox, the server checks if there is a message in the linked list for a particular receiver IP. If yes, then it removes it from the list.
7. The server sends that message to the destined receiver IP.
8. The message is then decrypted at the receiver’s end using client (C2) private key to obtain original message.

Output Snapshots and Explanations:

<table>
<tr><td>
RSA key pair generated and saved successfully.
Listening ...</td><td>Server-side program is executed first and it is waiting for clients to join.</td></tr>
<tr><td>
Public file: ./CA/public_key_123.pem
Private file: private_key_123.pem
RSA key pair generated and saved successfully.
My IP: 123
Connected to Server!
IP Address sent to server.</td><td>Client 1 generates its private key and public key pair and connects to the server with IP 123.</td></tr>
<tr><td>
Public file: ./CA/public_key_345.pem
Private file: private_key_345.pem
RSA key pair generated and saved successfully.
My IP: 345
Connected to Server!
IP Address sent to server.</td><td>Client 2 generates its private key and public key pair and connects to the server with IP 345.</td></tr>
<tr><td>
Hey! How are you?
Enter receiver IP: 345
Encrypting message ...
Encrypted text: 248E4ADDA56A55B387A1FF4395B675F6755C610ACC209
47FA2695E5C5F7B3E81D6C9B05F31DAC86A044D34453033C9D0104853EE26
099684C8C28A11B8D7389A
Message sent successfully!</td><td>Client 1 writes a message for client 2 and this message is encrypted using client2 ‘s public key and then sent</td></tr>
<tr><td>
Message received: 123|345|64|$◆J؍jU◆◆◆◆C◆◆u◆u\a
◆ ◆◆i^_{>◆◆щ_1◆◆jM4E03◆◆HS◆&   ◆◆◆◆◆◆8◆
Message inserted into Inbox of 345.

Message deleted from Outbox of 345
Message sent to 345</td><td>Server receives the message from client 1, acquires semaphore of client 2 and stores it in inbox of client 2 in a linked list format. When any message is in inbox, then first message is deleted from front referred to outbox and fetched by server and sent to particular client 2.</td></tr>
<tr><td>
Message received
Received Encrypted text: 248E4ADDA56A55B387A1FF4395B675F6755C
610ACC20947FA2695E5C5F7B3E81D6C9B05F31DAC86A044D34453033C9D01
04853EE26099684C8C28A11B8D7389A
Decrypting message ...
Received Message from 123 : Hey! How are you?</td><td>Client 2 receives the message and the IP address of the sender as well.</td></tr>
</table>

IV. Conclusion:

This project successfully addresses the critical need for secure and efficient communication between multiple clients over a network. By implementing end-to-end encryption using the RSA algorithm and incorporating a mailbox mechanism at the server, we have ensured the confidentiality of messages while maintaining scalability and efficiency. Through this implementation, we have contributed in providing a robust solution that prioritizes security without compromising on performance. As technology continues to evolve, the significance of secure communication will only grow, and our project stands as a testament to the ongoing efforts to safeguard sensitive information in an increasingly interconnected world.

V. Learning outcomes:

Proficiency in implementing encryption algorithms, specifically RSA, for ensuring data confidentiality in communication systems.
Understanding of the design and implementation of mailbox mechanism for efficient and scalable message passing between multiple clients and a server.
Understanding integrating cryptographic techniques into network protocols to provide encryption, balancing security with performance using synchronization mechanisms like semaphores.
Skills in identifying and addressing challenges related to secure communication, including message delivery optimization.

References

1. https://wiki.openssl.org/index.php/EVP_Symmetric_Encryption_and_Decryption
2. https://medium.com/hprog99/overcoming-message-delivery-challenges-in-distributed-systems-acomprehensive-look-at-outbox-and-a669e5f21898