CSE-539: Applied Cryptography
Bonus Project Report
Secure Messaging (Signal)

Kiran Kamalakar (1229582590)

Design

This project implements a secure communication system between three entities: Alice, Bob, and a Server. The system architecture follows a client-server model where:

- The server acts as a mediator or Key Distribution Center (KDC).
- Alice and Bob are clients who want to communicate securely.
- Communication between clients is secured by key exchange and encryption protocols mediated through the server.

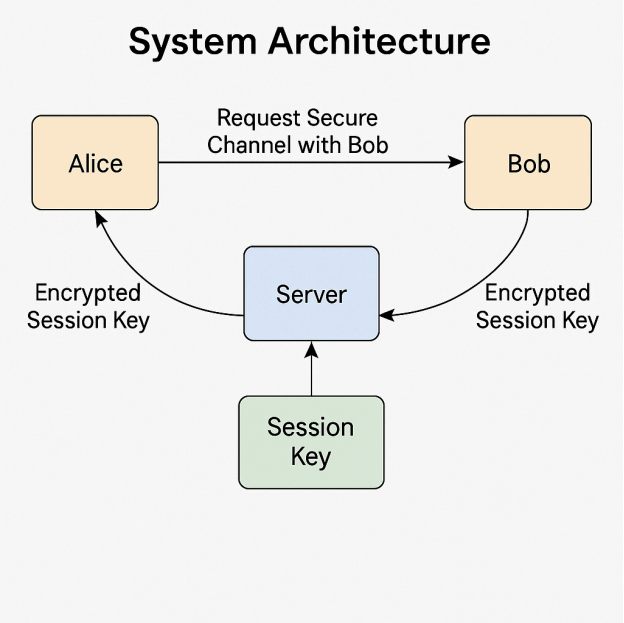

Figure 1: System Architecture Diagram

The overall flow:

1. Alice contacts the server to request a secure channel with Bob.

2. The server generates session keys and distributes them securely using pre-shared keys.
3. Alice and Bob establish communication using the provided session key.

This resembles a simplified version of Needham-Schroeder or Kerberos-like protocol for authentication and secure key exchange.

Security Intuition

The protocol's security relies on:

- Symmetric encryption (AES or similar) for the confidentiality of messages.
- Use of session keys instead of reusing long-term keys to limit exposure if a key is compromised.
- The server acts as a trusted authority to issue keys and authenticate participants.

Key security mechanisms include:

- Preventing **replay attacks** by using session keys per session.
- Avoiding direct sharing of private keys between clients.
- Encapsulation of session keys within encrypted payloads sent individually to Alice and Bob.

The security assumption is that:

- Communication with the server is initially secure (protected by pre-shared keys).
- The server is not compromised (acts as a fully trusted authority).

Implementation

Server (server.py)

- Listens for client requests (likely via sockets).
- Receives Alice's request for Bob's communication.
- Generates a random session key.
- Encrypts the session key with Alice's and Bob's pre-shared keys individually.
- Sends encrypted session keys back to Alice and Bob.

Alice (alice.py)

- Initiates the request to the server, specifying Bob as the target.
- Receives encrypted session key from the server.
- Decrypts using her pre-shared key.
- Uses the session key to encrypt messages sent to Bob.

Bob (bob.py)

- Receives encrypted session key from the server.
- Decrypts using his pre-shared key.
- Uses the session key to decrypt messages from Alice.

Common implementations:

- Encryption is likely done via pycryptodome or a similar library.
- Messages are serialized using pickle or JSON over sockets.
- Random session keys are generated securely using get_random_bytes or equivalent.

Conclusion

The project demonstrates a practical implementation of a secure key distribution and communication system using symmetric key cryptography mediated by a trusted server.

It aligns with principles of cryptographic security for small-scale or educational purposes but would require additional protections (e.g., against MITM, replay attacks, server compromise) for deployment in real-world systems.