Problem Statement

Cryptography Simulation with OpenSSL Library

Cryptography is essential for securing communication in the digital world, involving techniques like encryption and decryption to protect data. Understanding these concepts is crucial for students, developers, and professionals in computer security.

The objective is to develop an interactive simulation of cryptographic techniques using C code and mbedTLS/OpenSSL, enabling users to understand and experiment with different aspects of cryptography, including encryption, decryption, key generation, and cryptographic algorithms.

Idea

The proposed solution enhances secure communication between a client and server by using advanced cryptographic techniques to ensure data confidentiality, integrity, and authenticity through the use of :

- **AES-GCM encryption**: Provides secure encryption with authentication to ensure data confidentiality and integrity.
- **Secure key exchange mechanisms**: Ensures the secure establishment of encryption keys between client and server.
- **Digital signatures**: Verifies the authenticity and integrity of the communicated data.

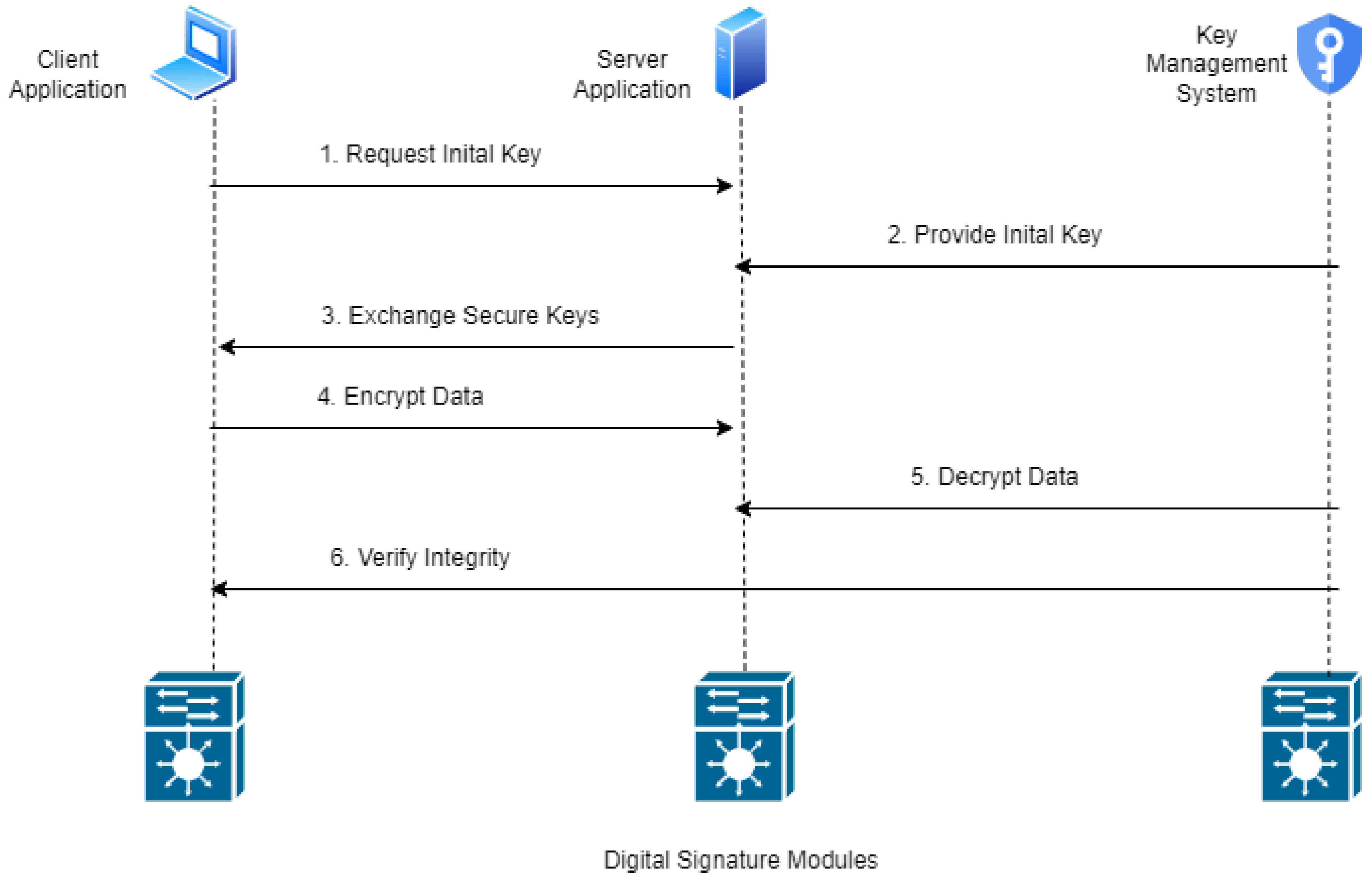
Features Offered

- AES-GCM encryption and decryption
- Secure key exchange
- Digital signature generation and verification
- Data integrity checks
- Initial key generation and distribution
- Secure communication setup between client and server
- Data encryption and transmission
- Data decryption and integrity verification on the receiver’s side

Process Flow

- **Initial Key Generation and Distribution**
  - Step 1: Generate cryptographic keys using secure algorithms (e.g., RSA, ECC).
  - Step 2: Distribute the public keys to clients through a secure and authenticated channel (e.g., via a Certificate Authority).
- **Secure Communication Setup**
  - Step 3: Client requests a secure communication session with the server.
  - Step 4: Server sends its public key to the client.
  - Step 5: Client uses the server's public key to encrypt a session key (symmetric key).
  - Step 6: Server decrypts the session key using its private key, establishing a shared secret for the session.
- **Data Encryption and Transmission**
  - Step 7: Client encrypts the data using the session key with AES-GCM encryption.
  - Step 8: Client sends the encrypted data along with an authentication tag to the server.
- **Data Decryption and Integrity Verification**
  - Step 9: Server receives the encrypted data and the authentication tag.
  - Step 10: Server decrypts the data using the session key.
  - Step 11: Server verifies the integrity and authenticity of the data using the authentication tag.

Architecture Diagram



Technologies used

OpenSSL

NASM

C++

Strawberry Perl

Visual Studio 2022 IDE

Team members and contribution

Nikita Palde:

Team Management

Functions :

- decryptAES_GCM256()
- checkCertificate()

.

Anushka Naik :

Functions :

- hmac_SHA256()
- deriveKey_HKDF_SHA256()
- encryptAES_GCM256()

Team members and contribution

Pragati Jadhav :

Functions :

- readRSAKeyFromFile
- signMessageRsa3072Pss, verifyMessageRsa3072Pss
- startdh

Digvijay Sonawane :

Project Report, Project work flow

Functions :

- getPublicKeyFromCertificate

Devansh Dubey :

Research, Installation of softwares on all laptops.

Conclusion

Our project successfully developed an interactive simulation of cryptographic techniques using the OpenSSL library. Despite only completing half of the intended functions, this achievement provided valuable insights into encryption, decryption, key generation, and cryptographic algorithms, enhancing our understanding and practical skills in computer security.