Intel Unnati Training
Summer Internship – 2024

Contents:

Team Name: TriNova

Team Members:

1. Maitri Rajesh Dulla
2. Ayushi Meshram
3. Sakshi Sunil Pawar

Project Overview: Cryptography Simulation with mbedTLS/OpenSSL

Objective:

The project aims to create an interactive cryptography simulation platform leveraging mbedTLS or OpenSSL libraries. Users will be able to establish secure connections between a server and a client, ensuring that only the respective key can decrypt or analyze the conversation, keeping third parties from accessing the communication.

Technologies Used:

- **Cryptographic Libraries:** OpenSSL, mbedTLS
- **Programming Languages:** C++
- **Software:** Visual Studio 2019/2022, Wireshark
- **System Resources:** Command Prompt

Challenges:

- **Key Management:** Creating a secure and user-friendly system for generating, distributing, and storing keys was essential for maintaining overall security.

- **Real-time Encryption and Decryption:** Implementing real-time encryption and decryption while maintaining responsiveness and efficiency, especially with large data volumes, was challenging.

- **Error Handling and Troubleshooting:** Debugging cryptographic operations required extensive testing and complex error-handling mechanisms.

- **Resource Management:** Efficiently managing system resources (CPU, memory) during encryption and decryption processes was crucial to avoid performance issues and ensure a smooth user experience.

- **Connectivity with OpenSSL:** Ensuring seamless integration with OpenSSL libraries required thorough understanding and management of various APIs and configurations.

- **Data Protection:** Robust encryption and secure storage solutions were necessary to protect user data from unauthorized access, especially during sensitive cryptographic operations.

- **Establishing Secure Connections:** Developing robust protocols for secure connections, including handling edge cases and potential security flaws, demanded meticulous attention.

Learning Outcomes:

- Participants gained a comprehensive understanding of cryptographic principles and secure communication by integrating OpenSSL libraries.
- They learned to handle cryptographic keys, create secure server-client connections, and perform real-time encryption and decryption.
- The project provided valuable skills in performance optimization, error management, user authentication, and adherence to industry standards, resulting in a robust and secure cryptographic simulation platform.

Introduction to Cryptography:

Cryptography is the science of securing information and communication using codes and ciphers to ensure that only intended recipients can decipher the data. It involves methods such as encryption (transforming readable data into an unintelligible format) and decryption (restoring the original data). Digital signatures, hashing algorithms, and key exchange techniques are used to ensure confidentiality, integrity, authenticity, and non-repudiation. Modern cryptography, rooted in ancient practices, employs advanced mathematical algorithms and computer science principles to protect digital data in various contexts, including private communications, data storage, and secure online transactions. Cryptography is the foundation of cybersecurity, continually evolving to combat new threats and vulnerabilities in the digital landscape.

Fundamentals of Cryptography:

- **Confidentiality:** Ensuring that data cannot be read by unauthorized parties.
- **Authenticity:** Confirming that the data originates from a trusted source.
- **Integrity:** Ensuring that the data has not been tampered with during transit or storage.
- **Non-repudiation:** Ensuring that the sender cannot deny the authenticity or validity of the data.

Exercises:

Exercise 1: Digital Certificates & Keys

1. Create a self-signed root certificate (rootCA.crt) with RSA key size of 3072, SHA384, and serial number 01.
2. Generate an RSA key pair of size 3072 with SHA384 for "Alice," sign with root CA, and set serial number 02.
3. Generate an RSA key pair of size 3072 with SHA384 for "Bob," sign with root CA, and set serial number 03.

Exercise 2:

1. Implement a crypto wrapper and ensure the crypto unit tests pass.
2. Secure the protocol using your crypto wrapper implementation.

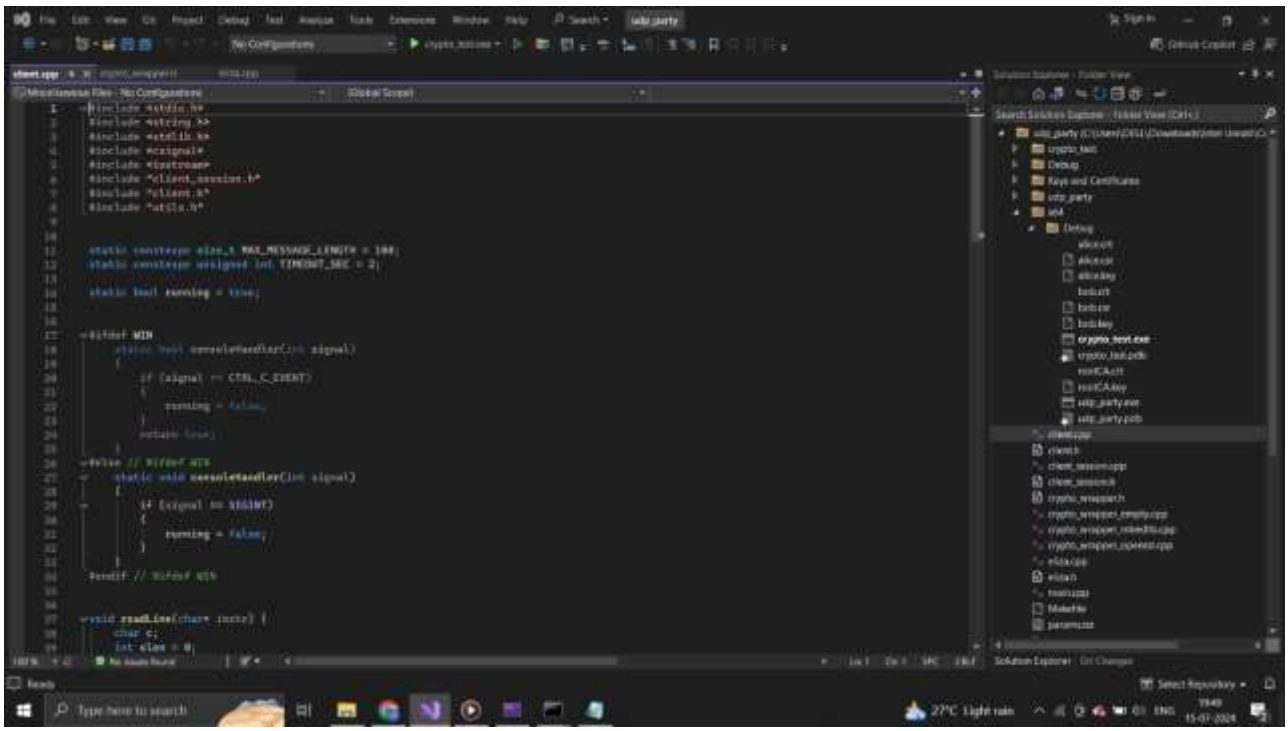
Crypto-wrapper Test Case:

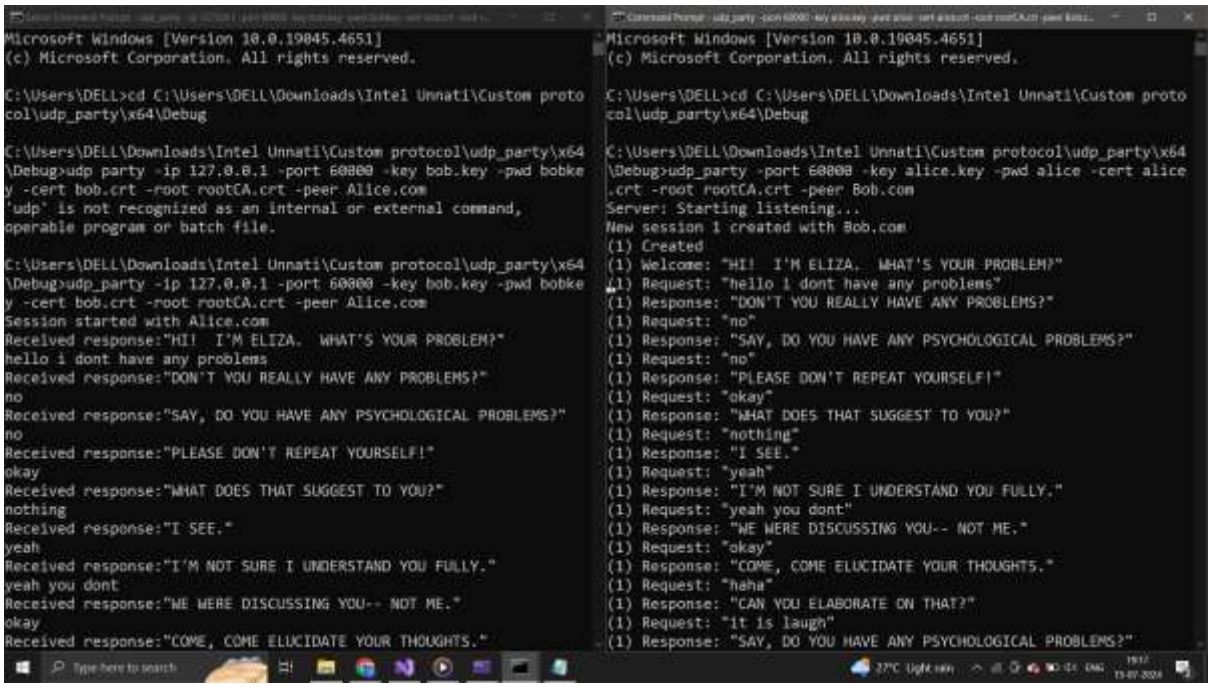
- **Client and Server Connection:** Establish a secure, encrypted connection between the client and server.
- **Encrypted Outputs:** Ensure data transmitted between client and server is encrypted.

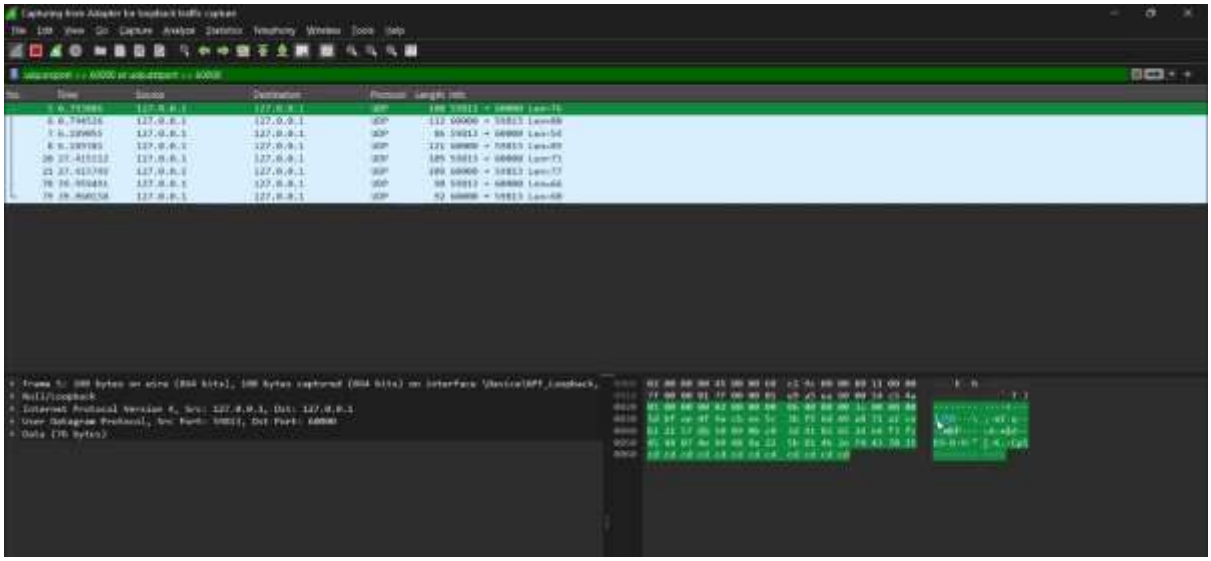
Applications:

- Secure communication
- Digital signatures
- Data encryption
- Secure email
- Blockchain and cryptocurrencies
- Secure file storage
- Authentication mechanisms
- Digital certificates
- Secure voting systems
- Secure messaging apps
- Access control systems
- Payment systems and online banking
- Virtual Private Networks (VPNs)
- Secure remote access
- Intellectual property protection

Outputs:







Conclusion:

Using OpenSSL or mbedTLS libraries, this interactive cryptography simulation platform offers a reliable method for creating secure connections between a client and a server. It ensures that all interactions are encrypted and protected from unauthorized access, only decryptable with the appropriate keys. This platform provides users with valuable practical experience and deep insights into the critical role of cryptography in securing digital communications, showcasing the importance of user-friendly design, secure API development, and staying abreast of the latest developments in cryptography.