Secure Messaging Client: User Manual

Student One
Student Two
Student Three
Student Four

October 20, 2025

Prepared for: Phase 2 Project - Code Theory and Cryptography

Contents

1 Introduction **2**

2 System Requirements **2**

3 Installation **2**

4 Running the Application **2**

5 Main Menu Options **2**
5.1 1. Create User 2
5.2 2. Send Message 3
5.3 3. View Messages 3
5.4 4. Verify Message 3
5.5 5. Show User Info 3
5.6 6. Demo Mode 3
5.7 7. Save User Keys 3
5.8 8. Load User Keys 3
5.9 9. Export Public Key 3
5.10 10. Import Public Key 3
5.11 11. Export Message Package 3
5.12 12. Import Message Package 4
5.13 13. Exit 4

6 Advanced Features **4**
6.1 Session Management 4
6.2 Export and Import 4

7 Troubleshooting **4**

8 Conclusion **4**

1 Introduction

The Secure Messaging Client is an educational tool designed to demonstrate secure communication principles using RSA for key exchange and AES-256-GCM for message encryption. This user manual provides step-by-step instructions on how to use the application, from installation to advanced features like exporting and importing keys and messages.

This application simulates encrypted messaging in a local environment and is intended for learning purposes. It does not support real network communication.

2 System Requirements

- Python 3.x (tested on Python 3.12.3 or similar) - The `cryptography` library (install via `pip install cryptography`) - No additional hardware requirements beyond a standard computer

3 Installation

1. Ensure Python 3.x is installed on your system. 2. Open a terminal or command prompt. 3. Install the required library:

```
pip install cryptography
```

4. Download or copy the `rsa.py` script to your working directory.

4 Running the Application

1. Navigate to the directory containing `rsa.py` in your terminal. 2. Run the script:

```
python rsa.py
```

3. The application will start and display the main menu.

Upon launching, you will see:

```
================================================================================
SECURE MESSAGING CLIENT
RSA-2048 Key Exchange + AES-256-GCM Encryption
================================================================================
```

Followed by the options menu.

5 Main Menu Options

The application uses a menu-driven interface. Enter the number corresponding to your choice.

5.1 1. Create User

- Creates a new user with a unique username and generates an RSA-2048 key pair. - Steps: 1. Enter the username when prompted. 2. The system will generate keys and display detailed key information (for educational purposes). - Note: Usernames must be unique. If the username exists, an error will be shown.

5.2 2. Send Message

- Sends an encrypted message from one user to another. - Steps: 1. Enter the sender's username. 2. Enter the recipient's username. 3. Enter the message text. - The system will establish a session if none exists, encrypt the message, and store it locally. - Requirements: Both users must exist, and the recipient must have a private key (i.e., not imported via public key only).

5.3 3. View Messages

- Displays all sent messages with cryptographic details. - Shows plaintext, encrypted data (IV, ciphertext, tag), and encryption info. - Useful for reviewing message history.

5.4 4. Verify Message

- Decrypts and verifies a specific message. - Steps: 1. Enter the 0-based message index (from View Messages). - Displays decrypted text and checks if it matches the original.

5.5 5. Show User Info

- Displays information about a user, including key details and active sessions. - Steps: 1. Enter the username.

5.6 6. Demo Mode

- Runs a predefined demonstration. - Creates users "Alice" and "Bob". - Sends sample messages. - Displays the message history. - Ideal for first-time users to see the system in action.

5.7 7. Save User Keys

- Saves a user's full key pair to a JSON file. - Steps: 1. Enter the username. 2. Enter the filepath (e.g., alice_keys.json).

5.8 8. Load User Keys

- Loads a user's keys from a JSON file. - Steps: 1. Enter the filepath. - The user will be added if not already present.

5.9 9. Export Public Key

- Exports a user's public key to a JSON file for sharing. - Steps: 1. Enter the username. 2. Enter the filepath (e.g., alice_public.json).

5.10 10. Import Public Key

- Imports a public key from a JSON file, allowing sending messages to that user. - Steps: 1. Enter the filepath. - Note: Imported users cannot receive/decrypt messages without their private key.

5.11 11. Export Message Package

- Exports an encrypted message package for sharing. - Steps: 1. Enter the 0-based message index. 2. Enter the filepath (e.g., message_package.json). - Includes the encrypted message and session key (encrypted for the recipient).

5.12 12. Import Message Package

- Imports and decrypts a shared message package. - Steps: 1. Enter the filepath. 2. Enter your username (as the recipient). - Adds the message to local history if successful.

5.13 13. Exit

- Shuts down the application.

6 Advanced Features

6.1 Session Management

- Sessions are automatically established when sending the first message. - Session keys are reused for efficiency in subsequent messages.

6.2 Export and Import

- Use export/import for simulating sharing keys or messages between instances. - Public key import allows one-way communication. - Message packages enable offline message transfer.

7 Troubleshooting

- **User not found**: Ensure users are created or imported correctly. - **No private key**: Imported public-key-only users cannot decrypt; create the user locally for full functionality. - **Invalid index**: Check message indices from View Messages. - **File not found**: Verify filepaths for load/export operations. - **Decryption failed**: Ensure session keys match and data is not tampered.

If issues persist, review console output for detailed error messages.

8 Conclusion

This Secure Messaging Client provides a hands-on way to explore cryptography. By following this manual, you can create secure sessions, exchange messages, and manage keys effectively. For technical details, refer to the Technical Documentation.