COMPE560-01:
Computer Data Networks, Spring 2025
HW1: UDP-based chat application.

Prepared by
Krishna Vardhan Nagaraja – 132711056
E-mail: knagaraja3869@sdsu.edu
Electrical and Computer Engineering
San Diego State University

OBJECTIVE:

The goal of this project is to build a UDP-based chat application with a graphical user interface (GUI). The application uses RSA for key exchange, AES for message encryption, and HMAC (Hash-based Message Authentication Code) to ensure message integrity and authenticity. The GUI is developed using PySide6.

Tech Stack:

- **Python:** Core programming language
- **Pyside6:** Python library for developing GUI
- **Socket:** Built in python feature for socket programming
- **Pycryptodome:** Python's cryptographic library for implementing RSA, AES and HMAC.

Cryptographic design:

1) **RSA Key exchange:** Each clients generate an RSA key pair. The public is shared by the client to the server, which uses it encrypt the AES key. Then the AES key is sent to client.
2) **AES Key Encryption:** Once the AES key is sent the messages are encrypted to ensure confidentiality.
3) **HMAC:** Every message has a HMAC tag which is generated using AES key, which ensures message integrity and authenticity.

Application Design:

- **Server Side:** The server listens for UDP packets and maintains a list of client addresses to their corresponding AES and public RSA keys. When it detects a new client's public key, it generates a unique AES key which is encrypted with RSA. For the messages the server verifies the integrity with HMAC and decrypts the content with senders AES key. Then the message is rebroadcasted to all the clients that are connected after encrypting it with the respective client's AES key.
- **Client Side:** The client sends its public RSA key to server and receives an AES key with RSA encryption. It uses this AES key to encrypt the messages thar are to be sent along with HMAC. For incoming messages, the client decrypts it using

AES key and messages are displayed on a GUI. A reliability feature is also added to send messages in case of failures.

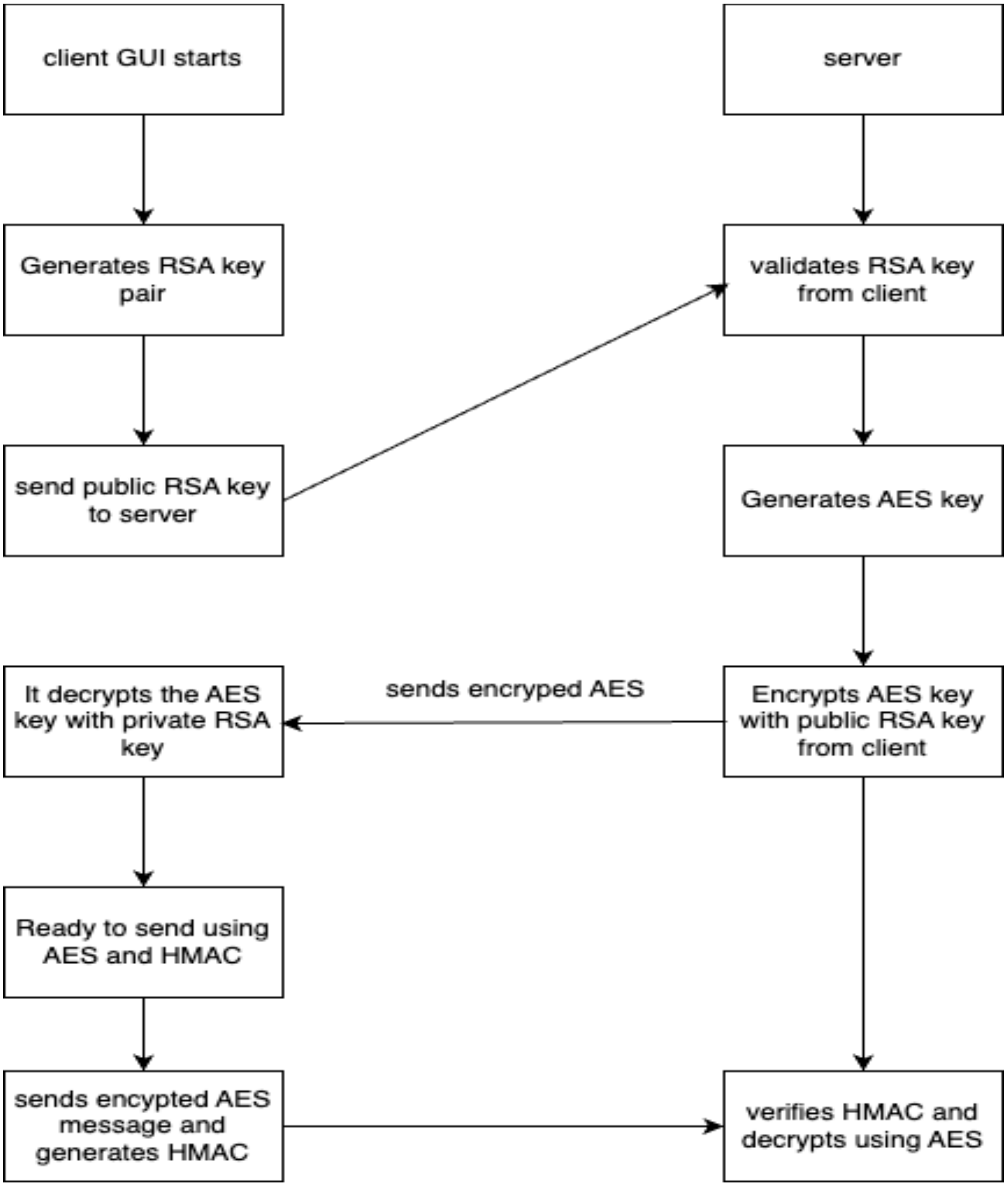
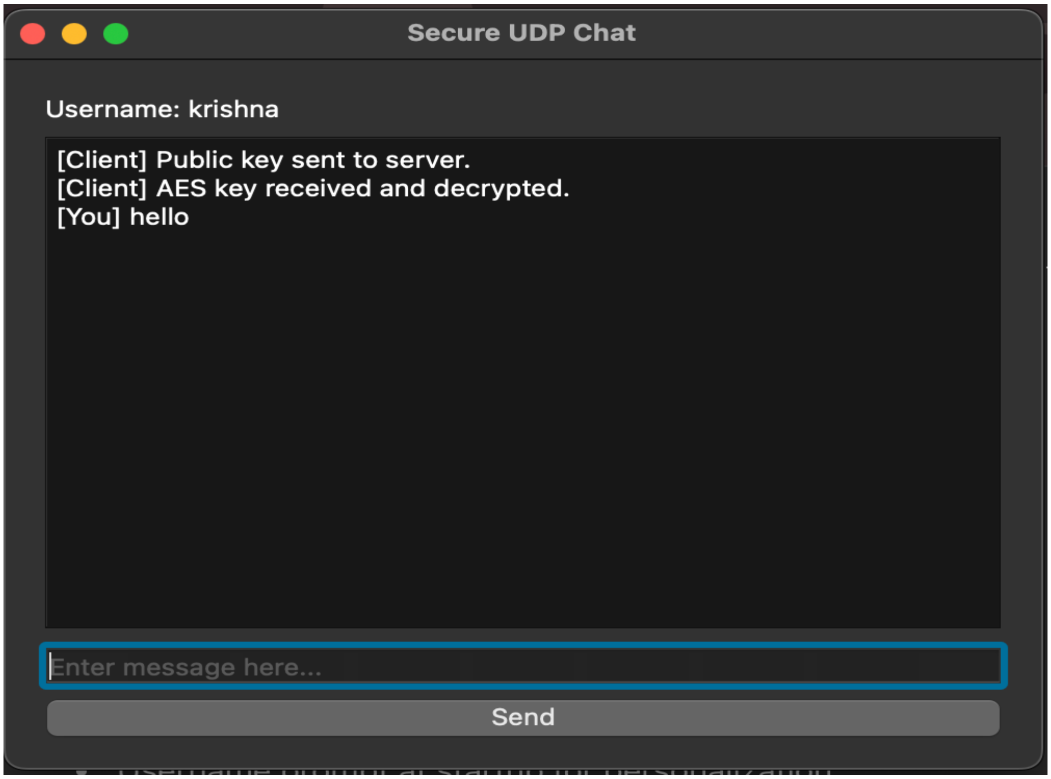

client GUI starts
server
Generates RSA key pair
validates RSA key from client
send public RSA key to server
Generates AES key
It decrypts the AES key with private RSA key
sends encryped AES
Encrypts AES key with public RSA key from client
Ready to send using AES and HMAC
sends encypted AES message and generates HMAC
verifies HMAC and decrypts using AES

Figure 1 Flow chart for client and server

GUI Design: A GUI built using Pyside6 is used for displaying the messages on client's side. It has a text display area which is read only and a text input field with send button for transmitting messages.



Username: krishna

```
[Client] Public key sent to server.
[Client] AES key received and decrypted.
[You] hello
```

Figure 2 GUI For the client

Conclusion: The chat application above provides secure UDP chat application with cryptographic techniques and basic user-friendly interface. The use of RSA, AES and HMAC increases integrity and authenticity of the messages. Along with the above features the GUI makes it suitable for real time messaging application.