**Intern Details**  
**Name:** Swarn Ranjan  
**Internship Duration:** 06th May 2025 – 30th June 2025  
**Institution:** Kalinga Institute of Industrial Technology  
**Mentor:** Dr. Tarun Yadav, Scientist-F, SAG, DRDO, New Delhi  
**Reporting Period:** Week 6 (05/06/2025 – 19/06/2025)

**1. Topics Covered**

This week focused on enhancing the secure chat application by introducing RSA-based message authentication and automatic RSA key pair management. Notable improvements include:

* Automatic RSA key generation for new users if key files are missing.
* Secure public key transmission to the server upon client connection.
* Message and file integrity verification using SHA-256.
* Digital signature verification on the server to authenticate message origin.
* Modular improvements in client.py for usability and trust establishment.
* Persistent key storage in .pem format per user.

**2. Resources Utilized**

 PyCryptodome documentation – RSA key generation and signature APIs

 PyQt5 documentation – GUI widget layout, events, threading

 NIST SHA-256 specification

 Python socket and threading modules

 Stack Overflow and GitHub for implementation strategies

**3. Key Learnings**

**** RSA **Key Lifecycle**: Understood how to generate, store, and use RSA key pairs for signing and verifying messages.

 Signature **Authentication**: Implemented and tested digital signatures to validate message origin and detect tampering.

 Hybrid **Cryptographic Systems**: Combined AES (confidentiality), SHA-256 (integrity), and RSA (authenticity) effectively.

 GUI **Engineering**: Learned how to build responsive GUI applications using PyQt5 to interface with backend logic.

 Usability **and UX**: Improved overall user experience by allowing chat clients to communicate with usernames and structured interactions.

**4. Project Status**

The secure chat system is now feature-complete for core objectives:

* Secure text and file transfer using **AES-CBC encryption**
* SHA-256 hashing to **verify message and file integrity**
* **RSA digital signatures** to ensure message authenticity
* **Automatic RSA keypair generation** on client side
* **Graphical interface** using PyQt5 for:
  + Connecting with username
  + Sending/receiving encrypted messages and files
  + Real-time updates in a chat-style interface
* Server validation for public keys and signature checks
* Refactored architecture with clear modular files:

server.py, client\_logic.py, crypto\_utils.py, chat\_gui.py

**5. Code Snippets**

Automatic RSA Key Generation:

 if not (os.path.exists(privkey\_path) and os.path.exists(pubkey\_path)):

            print(f"[KEYGEN] RSA keys not found for '{USERNAME}'. Generating...")

            key = RSA.generate(2048)

            with open(privkey\_path, "wb") as priv\_file:

                priv\_file.write(key.export\_key())

            with open(pubkey\_path, "wb") as pub\_file:

                pub\_file.write(key.publickey().export\_key())

            print(f"[KEYGEN] Keys saved as '{privkey\_path}' and '{pubkey\_path}'.")

Digital Signature Function:

def sign\_data(data: bytes, private\_key\_path: str) -> bytes:

    key = RSA.import\_key(open(private\_key\_path).read())

    h = SHA256.new(data)

    signature = pkcs1\_15.new(key).sign(h)

    return signature

Signature Verification:

def verify\_signature(data: bytes, signature: bytes, public\_key\_path: str) -> bool:

    key = RSA.import\_key(open(public\_key\_path).read())

    h = SHA256.new(data)

    try:

        pkcs1\_15.new(key).verify(h, signature)

        return True

    except (ValueError, TypeError):

        return False

PyQt Message Handler:

 def send\_message(self):

        if not self.chat\_client:

            return

        message = self.message\_input.text().strip()

        recipient = self.recipient\_input.text().strip()

        if message and recipient:

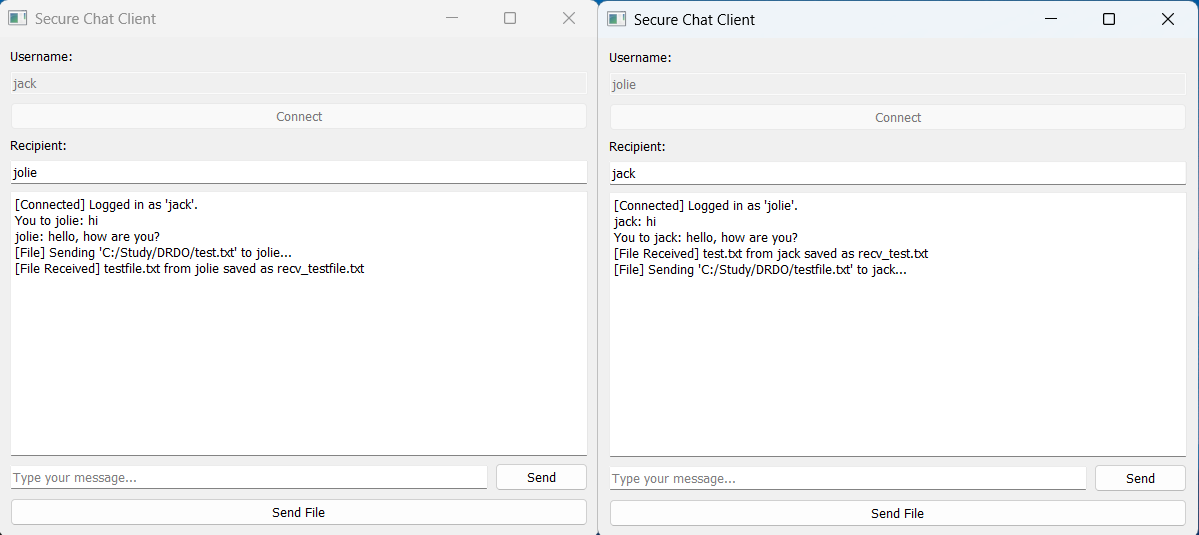
            self.chat\_client.send\_message(recipient, message)

            self.chat\_log.append(f"You to {recipient}: {message}")

            self.message\_input.clear()

**6. Outputs**

GUI output:



File outputs: A black screen with white text

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.

(Swarn Ranjan)