Information Security Lab

Secure File Transfer System

Project Report

Section: A

Submitted by:

Fiza Manzoor 2023-CS-612

Samar Noor 2023-CS-630

Submitted to:

Mam. Shanfa Irum

Dated: 4th May 2025

Department of Computer Science

University of Engineering and Technology Lahore, New Campus

Introduction:

The Secure File Transfer System addresses the growing need for secure data exchange over untrusted networks, where traditional file transfer methods like FTP expose sensitive information to interception and unauthorized access. This project leverages advanced encryption, Virtual Private Network (VPN) tunneling, and real-time secure transfer to ensure confidentiality, integrity, and visibility during file transfers. Designed as a Stream lit-based application, the system enables users to authenticate securely, encrypt files, transfer them over a VPN using Transport Layer Security (TLS), and monitor network traffic, all within a user-friendly interface. The primary motivation is to provide a practical solution for secure file sharing between two PCs in a controlled environment, addressing modern cybersecurity challenges such as data breaches and eavesdropping.

Objectives:

The Secure File Transfer System was developed with the following key objectives to ensure a robust and secure file-sharing solution:

- Secure user authentication using RSA digital signatures to verify user identities and prevent unauthorized access to the system.
- Enable file encryption and decryption with AES-256, ensuring data confidentiality during storage and transfer, with RSA facilitating secure key exchange.
- Facilitate secure file transfers over a Virtual Private Network (VPN) using Transport Layer Security (TLS) to protect data integrity and privacy between two PCs on port 8444.

Features:

The core features of the Secure File Transfer System, each designed to enhance security and usability is described below:

1. **User Authentication:** This feature authenticates users using RSA digital signatures, ensuring only authorized individuals can access the file transfer functionality.

   Functions:

   - init_db(): Initializes the SQL Server database with the Users table for storing usernames and public keys.
   - generate_keys(): Generates RSA key pairs and saves them as PEM files for user registration.
   - load_public_key(): Loads the public key from a PEM file for signature verification.
   - sign_message(): Generates a signature for login using the private key.
   - verify_signature(): Verifies the signature during login.

Role: These functions handle user registration and login authentication using RSA digital signatures, integrated into the "Login" and "Admin Login" pages.

Error Handling: Captures pyodbc.Error for database issues and displays specific error messages.

2. **File Encryption and Decryption:** This feature allows users to encrypt and decrypt text files using AES-256 in CBC mode, ensuring file confidentiality with a 32-byte key, supported by RSA for secure key exchange.

   Functions:

   - encrypt_file(content, key): Encrypts file content with AES-256 in GCM mode.
   - decrypt_file(content, key): Decrypts file content with the corresponding AES key.

Role: These functions enable secure file processing in the "File Processing" section of the "File Operations" page.

Error Handling: Captures pyodbc.Error for database issues and displays error messages.

3. **File Transfer with VPN and TLS:** This feature enables secure file transfer over a VPN between PC 1 and PC 2 on port 8444, using TLS for end-to-end encryption, with OpenVPN managing the VPN tunnel.

   Functions:

   - start_vpn_client(): Launches OpenVPN with client.ovpn.
   - stop_vpn_client(): Terminates the OpenVPN.
   - TLS Configuration: Enforces TLS 1.2 minimum for security.

Role: These functions manage the VPN tunnel, which is part of the "File Transfer" section, ensuring secure transfer with TLS.

Error Handling: Catches ssl.SSLError for TLS issues and socket.error for connection.

4. **Additional Functions:**

   - generate_tls_certificates(): Generates self-signed TLS certificates for secure file transfer.
   - get_db_connection(): Establishes the SQL Server connection.
   - load_css(): Applies custom CSS for styling the UI.

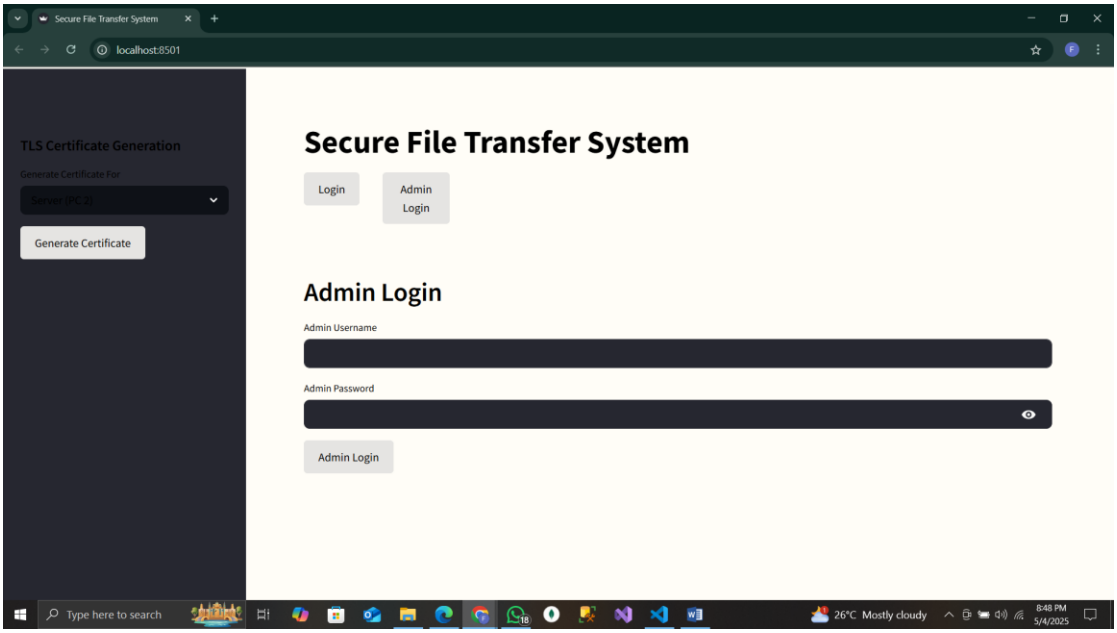
5. **Server side operation:**

   - The terminal running the server-side script (server.py) on PC 2.
   - The server receives the file from the pc1.

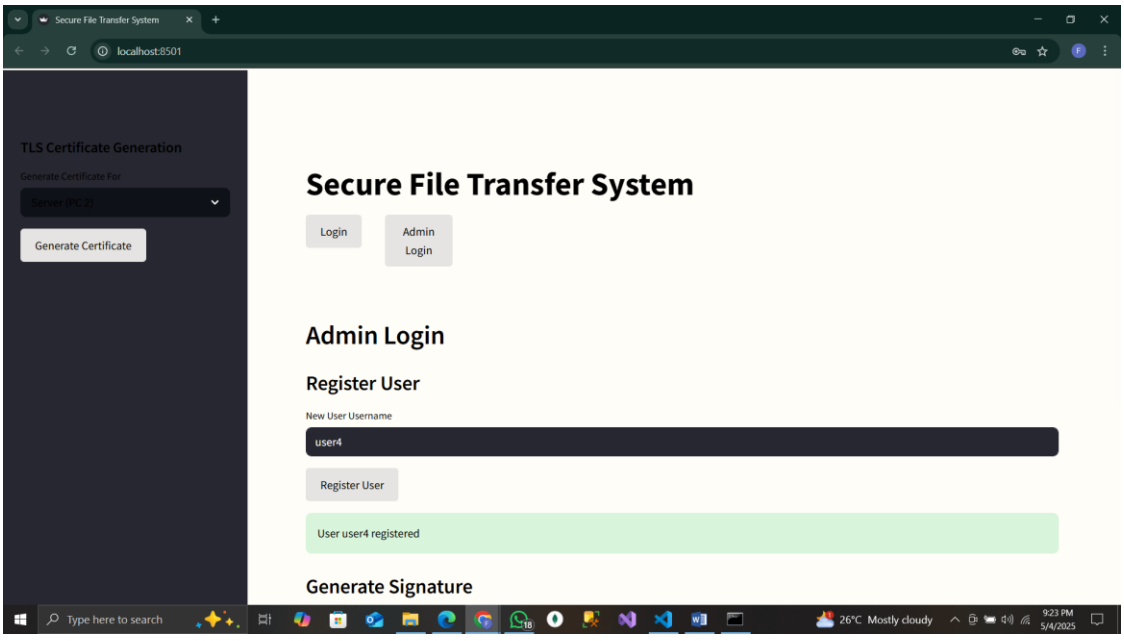
- After receiving a file from the pc1, it saves the file as received_file.txt in the same directory as server.py.

Visualizations:

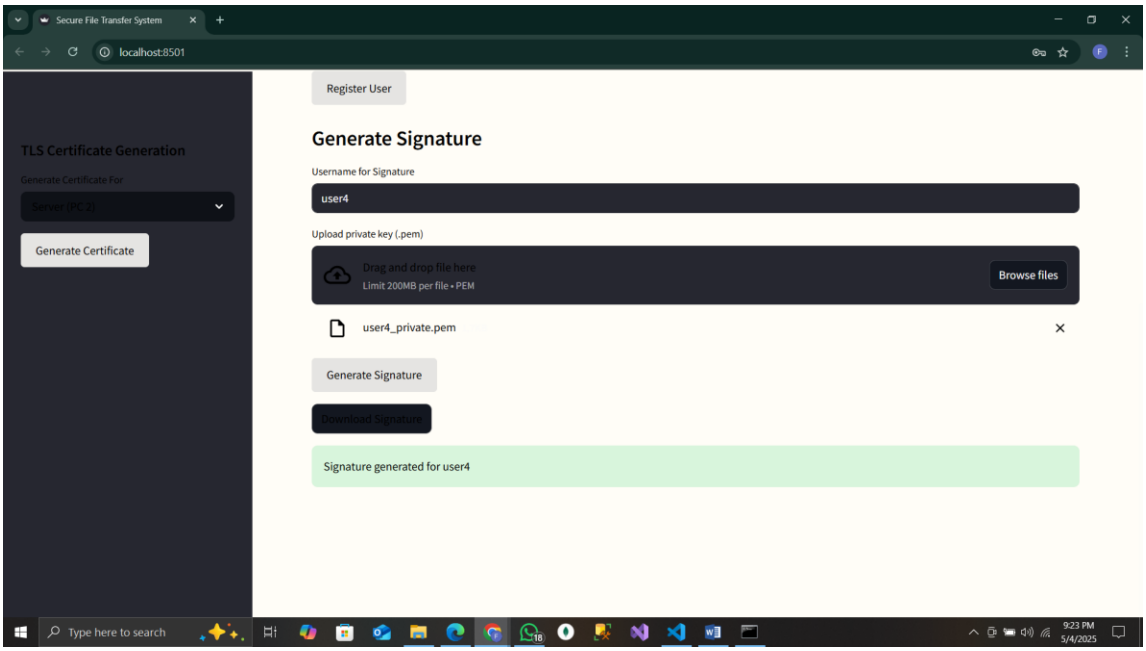
1. Admin login page:



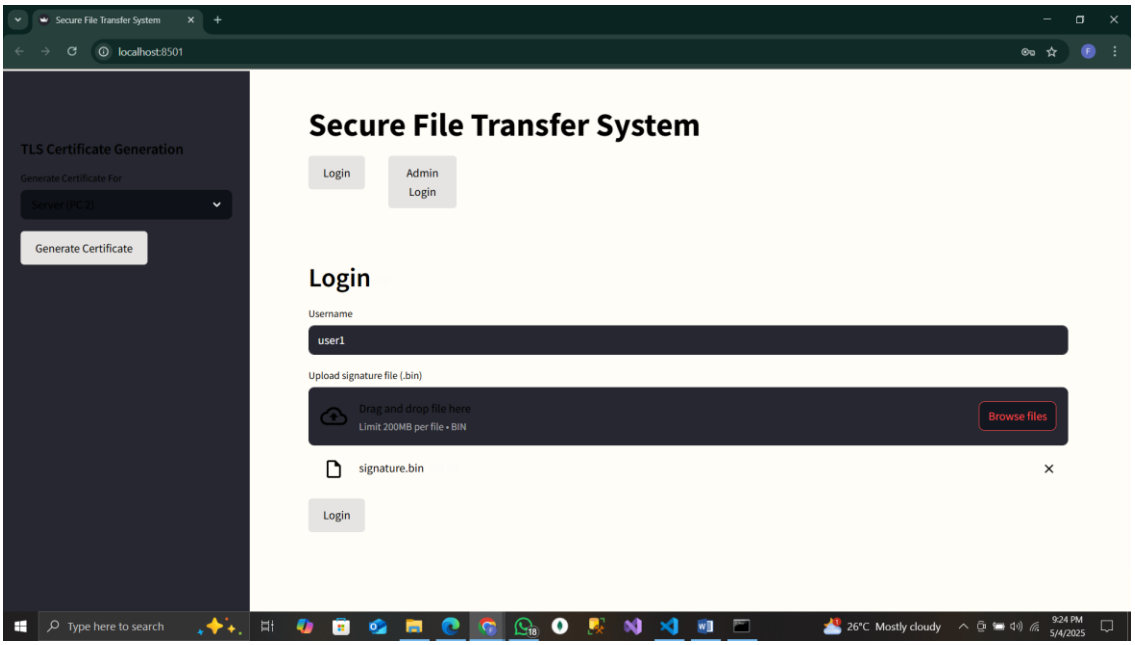
2. Register user: Admin register the user after logged in successfully.



3. Generate Signature: Generating signatures requires the private key of user.

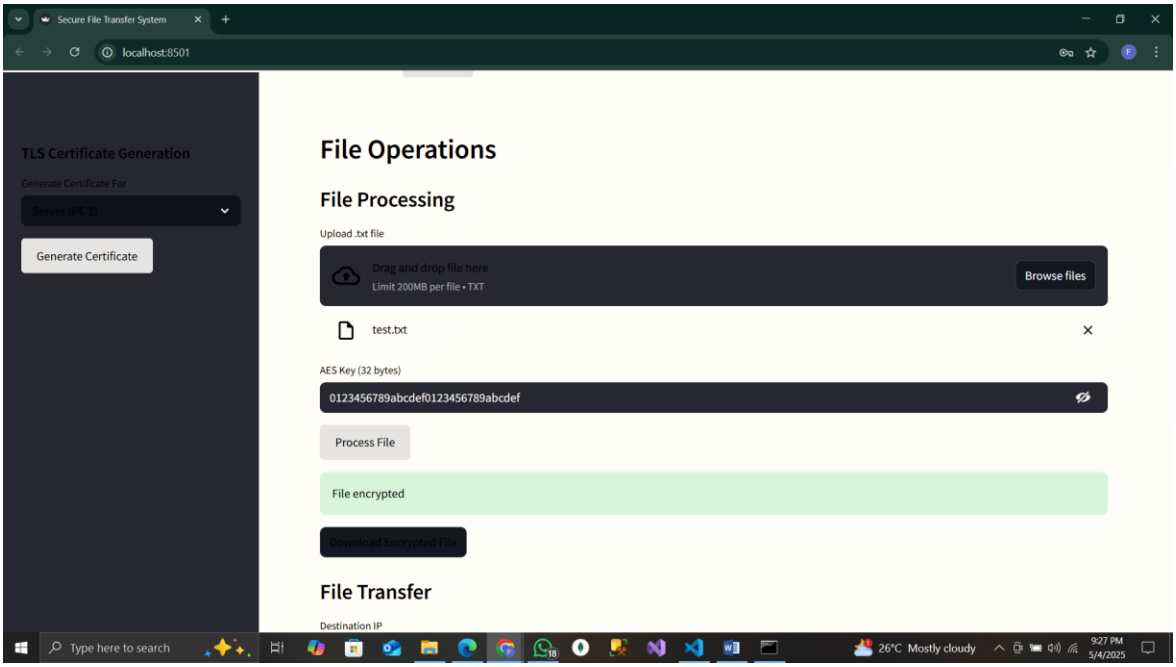


4. User login: User can logged in using his username and signature.

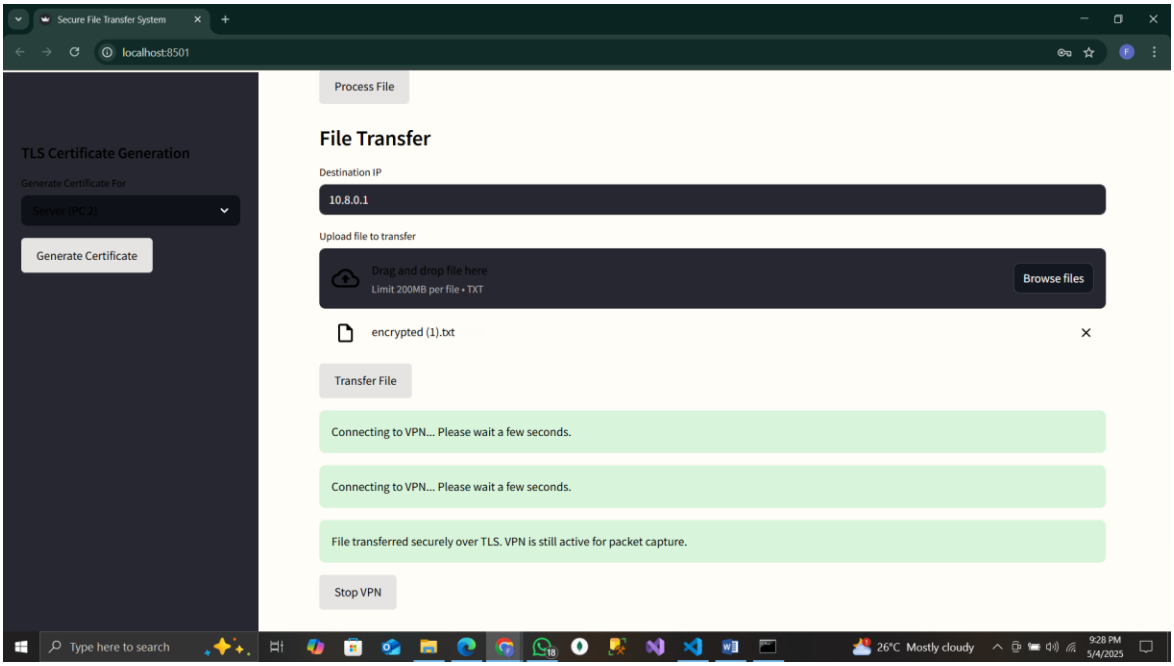


5. File Operations:

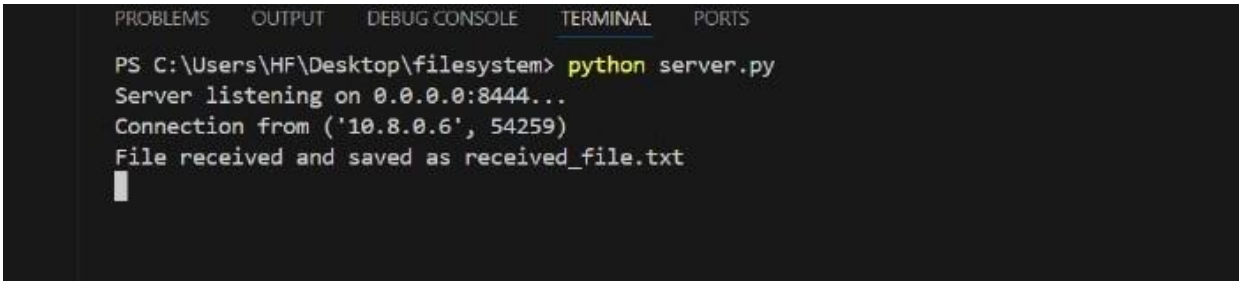
    - File processing: User enter the .txt file and the 32 bit key to encrypt or decrypt the file.



- File transfer: User enters the destination pc's IP and the file and clicks the Transfer file button.



6. Server side: File is received on the server pc and saved as received file in the directory.


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\HF\Desktop\filesystem> python server.py
Server listening on 0.0.0.0:8444...
Connection from ('10.8.0.6', 54259)
File received and saved as received_file.txt