SECURITY OVERVIEW DOCUMENT

Project: Secure File Sharing System

Developer: Vaibhav Malhotra

Internship Program: Future Interns – Cyber Security Task 3

Date: 28/07/25

Track Code: FUTURE_CS_03

Tools: Python, Flask, PyCryptodome, HTML, CSS

OBJECTIVE:

The goal of this task3 was to build a Secure File Sharing System that supports:

1. File upload and download by users after Encryption.
2. Use of AES Encryption Cipher Block chain.
3. Secure Download of Decrypted Files.

ENCRYPTION ALGORITHM USED: AES (Advanced Encryption Algorithm)

- AES is **a symmetric key encryption algorithm** used worldwide to secure data. It was established by the **U.S. National Institute of Standards and Technology (NIST)** in 2001, replacing the older DES **(Data Encryption Standard)** due to its vulnerabilities. **It converts plain texts to Cipher text using a secret key.**

- I used **AES-128-CBC-mode** (128 bit size, cipher block chaining) via **PyCryptodome Library in Python**

- AES is widely adopted across almost every industry where data needs to be kept private or secure. Here are the most common use cases:

1. Operation System & Disk Encryption
2. Internet Communication
3. Messaging Apps
4. File and Cloud Storage
5. Wi-Fi Security
6. Banking and Financial Services
7. Military Services
8. Backup and Archival Tools
9. Embedded Systems & IoT Devices

File Handling Process:

- Users upload the file via Front-End to Encrypt it.

- The file is Encrypted using AES Encryption algorithm and uploaded in (.enc) format.

- User can download the encrypted (.enc) file after the file has been uploaded directly from the link.

- To decrypt and download the file -> user has to copy the (.enc) file and paste it into the decrypt input field. Then the decrypt file gets downloaded (.dec).

SECURITY CONSIDERATION:

1. Files are never stored in plain text on the server.
2. The encryption key is not expose to users or stored alongside file.
3. HTTPs is recommended for deployment to secure file transfers.

FUTURE IMPROVEMENTS:

- A **Secure Login/authentication** system for users.
- Add **File size Limits.**
- Store Encryption keys in dedicated key vault.
- Enable file **Auto-Deletion** after a set time for added security.

SUMMARY:

- This File sharing system Demonstrates how encryption algorithms ensure **confidentiality of a file**.

- This project also demonstrates how **encryption and secure file storage** can be integrated into a web app.