Cyber Security Internship - Task 2 Report

Intern Name	Jagdish Zate
Track Code	CS
Task Number	03
Internship Domain	Cyber Security
Task Title	Secure File Sharing System
GitHub Repository	https://github.com/JagdishZate400/FUTURE_CS_01.git

1. Objective

The objective of this task is to design and implement a file upload/download portal with built-in AES encryption. The goal is to ensure all files are secure at rest and in transit, simulating a secure file-sharing platform. This task emphasizes the implementation of encryption algorithms, key management, and secure file handling techniques.

2. Tools and Technologies Used

Python – Core programming language

Flask – Lightweight web framework for Python

PyCryptodome – Python library for AES encryption

HTML (Jinja2) – Frontend templating

Postman / Web browser – API testing and file handling

Git & GitHub – Version control and repository hosting

3. Environment Setup

Flask environment set up using pip to install required dependencies.

Project directory includes encrypted file storage (uploads/) and decrypted file handling (decrypted/).

All encryption logic is implemented in app.py.

Encryption key is predefined for demo purposes and should be secured via environment variables in production.

AES (CBC mode) implemented with dynamic IVs and proper padding/ unpadding logic.

4. Application Workflow

File Upload

User selects a file and uploads it through the frontend.

File content is encrypted using AES (CBC mode) with a random IV.

Encrypted file is stored with .enc extension in the uploads/ directory.

File Download

When user downloads an .enc file, it is decrypted on the server.

Original (decrypted) file is temporarily saved and sent back to the user for download

5. Security Mechanism

Element Details

Encryption AES – 128 bit, CBC mode IV Handling 16-byte randomly generated IV per file, prepended to ciphertext

Key Management Static key for demo (recommend using environment variable)

Padding SchemePKCS7 (handled via PyCryptodome utilities)

Transport Security Assumes HTTPS when deployed in production

6. Outcome & Learning

This task deepened understanding of log analysis and SOC workflows. Gained hands-on experience with Splunk, alert filtering, and reporting. Developed a structured approach to real-time security monitoring and response.

7. Deliverables

• GitHub Repository: https://github.com/JagdishZate400/FUTURE_CS_02 (add once created)

Security Report:

This report summarizes all encryption logic and secure file handling approaches

8. Task Reference

• Task Page: https://futureinterns.com/cyber-security-task-3/
• Internship Site: https://futureinterns.com

9. Conclusion

This task was successfully completed with all core objectives achieved. It showcases applied knowledge in:

Web development,

Data security,

And Cryptographic implementation.

All deliverables have been submitted, and the solution is production-ready with minor improvements (e.g., key management and HTTPS enforcement).

Appendix: Screenshots

1. Flask App Home Screen
2. File Upload Interface
3. Encrypted File Preview
4. Decrypted File Download
5. Encrypted .enc Storage Confirmation