Secure File Storage System with AES-256 Encryption

1. Introduction

In the modern digital era, sensitive data such as personal documents, financial records, academic files, and business information are frequently stored and shared electronically. Unauthorized access, data breaches, and file tampering pose serious security risks. Ensuring confidentiality, integrity, and secure access control of stored files is therefore essential.

Encryption is one of the most effective techniques used to protect sensitive information. This project focuses on developing a Secure File Storage System that encrypts and decrypts files using Advanced Encryption Standard (AES) technology. The system ensures that files remain unreadable to unauthorized users while maintaining integrity verification through hashing mechanisms.

The primary objective of this project is to implement a lightweight, local file encryption system using Python that demonstrates practical cryptographic principles in action.

2. Abstract

The Secure File Storage System is a Python-based application designed to encrypt and decrypt files securely using AES-based symmetric encryption. The system uses the Fernet module from the cryptography library, which provides strong encryption and secure key management.

Before encryption, a SHA-256 hash of the file is generated to ensure integrity verification. Encrypted files are stored with a .enc extension, and metadata such as original file name, timestamp, and file hash is recorded in a JSON file. During decryption, the encrypted file is restored to its original format using the same encryption key.

The system demonstrates practical implementation of cryptographic concepts including symmetric encryption, hashing, key management, and secure file handling. It runs entirely in a local environment without requiring network access, making it secure and easy to deploy.

3. Tools Used

The following tools and technologies were used to develop the project:

- Python 3.x – Core programming language used for implementation.
- cryptography (Fernet module) – Used to perform AES-based symmetric encryption and decryption.
- hashlib (SHA-256) – Used to generate file hashes for integrity verification.
- JSON module – Used to store metadata securely.
- VS Code – Development environment.
- Command Line Interface (CLI) – Used for user interaction

4. Steps Involved in Building the Project

Step 1: Project Setup

Step 2: Key Generation

Step 3: File Hashing (Integrity Verification)

Step 4: File Encryption

Step 5: File Decryption

Step 6: Metadata Management

---

5. Security Features Implemented

- AES-based symmetric encryption (via Fernet)
- Secure key storage
- SHA-256 integrity verification
- Metadata logging
- Local file handling (no external exposure)

6. Conclusion

The Secure File Storage System successfully demonstrates the practical implementation of modern cryptographic principles. By combining AES-based encryption with SHA-256 hashing, the system ensures both confidentiality and integrity of stored files.

This project highlights the importance of secure key management and encryption in protecting sensitive data from unauthorized access. It also reinforces foundational cybersecurity concepts such as symmetric encryption, hashing, and secure storage mechanisms.

The system can be further enhanced by:

- Adding password-based key derivation
- Implementing a graphical user interface (GUI)
- Adding digital signatures
- Integrating tamper detection mechanisms

Overall, this project provides a strong practical understanding of file security and encryption techniques, making it highly relevant in cybersecurity and data protection domains.