**Encryption Methods and Key Handling**

*A Comprehensive Overview of Data Protection Through Cryptographic Techniques*

Prepared by: Ankit Roy  
Date: June 28, 2025

**1. Introduction**

**In modern digital environments, the secure transfer and storage of sensitive data is critical. This internship project focuses on building a secure file sharing system that applies robust encryption techniques to protect data confidentiality and integrity during both storage and transmission. The system leverages Advanced Encryption Standard (AES) for file encryption and RSA public-key cryptography for secure key exchange, implementing a hybrid encryption model commonly used in real-world applications such as TLS/SSL and S/MIME.**

**2. Project Objectives**

**The project is designed to simulate secure data sharing between users in sectors such as healthcare, legal, finance, and enterprise IT, where data protection is mandatory. The key objectives include:**

* **Developing a web application for secure file upload and download**
* **Applying AES encryption to protect data at rest and in transit**
* **Implementing RSA to securely exchange AES session keys**
* **Ensuring strong key generation, storage, and management**
* **Creating an intuitive user interface for seamless file operations**
* **Validating file integrity and ensuring end-to-end encryption**
* **Documenting system architecture and encryption workflows**

**3. System Architecture and Encryption Workflow**

**3.1 Upload Process**

* **A random AES-256 symmetric key is generated per upload session.**
* **The file is encrypted using the AES key before storage.**
* **The AES key is encrypted using the recipient's RSA public key.**
* **Both the encrypted file and the encrypted AES key are stored securely on the server.**

**3.2 Download Process**

* **The user retrieves the encrypted AES key and file from the server.**
* **The AES key is decrypted using the user's private RSA key.**
* **The file is then decrypted using the decrypted AES key and delivered to the user.**

**4. Encryption Techniques Used**

**4.1 Symmetric Encryption (AES)**

* **Used for encrypting the file content.**
* **Offers fast and efficient processing.**
* **Key Length: 256 bits for maximum security.**

**4.2 Asymmetric Encryption (RSA)**

* **Used for encrypting the AES session key.**
* **Ensures secure key exchange without requiring prior sharing.**
* **Enhances access control by binding file access to the private key owner.**

**4.3 Hybrid Encryption Model**

* **Combines AES and RSA to provide both performance and secure key exchange.**
* **Ensures forward secrecy by generating new AES keys for each session.**

**5. Key Management Strategy**

**5.1 Key Generation**

* **AES keys are generated using cryptographically secure libraries (e.g., Python cryptography or secrets).**
* **RSA keys are generated per user during account setup.**

**5.2 Key Storage**

* **AES keys are never stored in plaintext.**
* **Encrypted keys are stored alongside encrypted files in the server/database.**
* **RSA private keys are securely stored on the client side.**

**5.3 Key Transmission**

* **Keys are transmitted over TLS-secured channels.**
* **AES keys are encrypted with RSA to prevent exposure in transit.**

**5.4 Key Rotation and Lifecycle**

* **AES keys are session-based and rotated per upload to ensure forward secrecy.**
* **RSA keys can be rotated periodically or upon detected compromise.**

**6. Security Measures Implemented**

* **✅ AES-256 encryption for file confidentiality**
* **✅ RSA public-key encryption for secure AES key exchange**
* **✅ TLS for secure communication between client and server**
* **✅ No plaintext key storage anywhere in the system**
* **✅ Role-based access control to restrict unauthorized access**
* **✅ Encrypted database storage for files and keys**
* **✅ Logging and monitoring for key access and file downloads**

**7. Skills Acquired**

* **Secure full-stack web development (Flask / Node.js)**
* **Applied cryptography (AES, RSA, hybrid encryption models)**
* **Secure file handling and data protection techniques**
* **Key management best practices (generation, encryption, rotation)**
* **Secure protocol usage (TLS, HTTPS)**
* **Documentation, version control (Git/GitHub)**

**8. Real-World Relevance**

**This project mirrors encryption workflows seen in:**

* **TLS/SSL (HTTPS) communication protocols**
* **Secure file transfer systems**
* **End-to-end encrypted messaging platforms**
* **Digital signing and secure document sharing applications**

**9. Conclusion**

**This project demonstrates how encryption, when properly implemented and managed, can safeguard sensitive data in a digital ecosystem. By leveraging hybrid encryption models, strong key management practices, and secure storage/transmission protocols, the file sharing system ensures confidentiality, integrity, and controlled access—essential pillars in the field of cybersecurity.**