SDU UNIVERSITY

SECURE FILE ENCRYPTION SYSTEM

Ernur Zinelov, Shyngyskhan Kumkay

PROBLEM STATEMENT

Why We Built This:

- Sensitive files need secure protection
- Users often store data without encryption
- Need for simple but strong encryption tool
- Must support key management & digital signatures

Ernur Zinelov, Shyngyskhan Kumkay

PROJECT OVERVIEW

What Our System Does:

- AES-256 file encryption & decryption

Secure symmetric encryption for fast protection of files.

- RSA digital signatures

Ensures data authenticity and integrity through signing and verification.

- SHA-256 hashing

Generates cryptographic hashes for data validation and integrity checking.

- Key Management System (AES/RSA keys)

Secure generation, storage, and rotation of cryptographic keys.

- Hybrid RSA + AES encryption

Combines RSA for secure key exchange and AES for efficient data encryption.

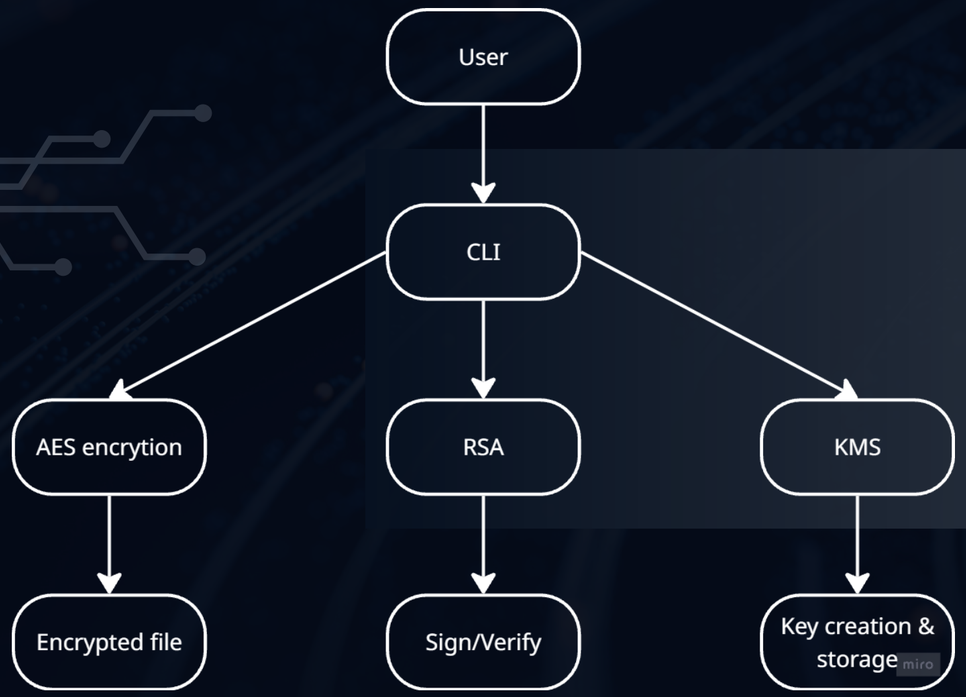
- Full CLI interface

Interactive command-line tool for all cryptographic operations.

SYSTEM ARCHITECTURE

- Modular Python architecture
- app/cli → user interface
- crypto/symmetric → AES + Scrypt KDF
- crypto/asymmetric → RSA + signatures
- kms/ → key generation & storage

Ernur Zinelov, Shyngyskhan Kumkay



DATA FLOW DIAGRAM

Ernur Zinelov, Shyngyskhan Kumkay

CRYPTOGRAPHIC COMPONENTS

- AES-256-GCM for encryption
- Scrypt KDF for password → key
- RSA-2048 for signatures & hybrid encryption
- SHA-256 for file hashing

Ernur Zinelov, Shyngyskhan Kumkay

SECURITY FEATURES

- Memory-hard KDF (Scrypt)
- GCM mode prevents tampering
- Signatures prevent forgery
- Keys separated in secure folders
- Hybrid encryption protects AES keys

LIVE DEMO OVERVIEW

Ernur Zinelov, Shyngyskhan Kumkay

CONCLUSION

Challenges faced Lessons learned

We dealt with cryptographic library complexity, key management issues, and several environment/venv errors. These challenges helped us understand the practical difficulties of building secure systems.

What We Learned

We gained hands-on experience with AES, RSA, hashing, KDFs, and modular system design. We also learned how to structure secure applications and write automated tests using pytest.

Final Outcome

The final system is fully functional, secure, and well-documented — providing encryption, signatures, key management, and hybrid encryption through a clean CLI interface.

SDU UNIVERSITY

THANK YOU FOR ATTENTION

Ernur Zinelov, Shyngyskhan Kumkay