The Academic
International Journal of Multidisciplinary Research

An Online Peer Reviewed / Refereed Journal
Volume 4 | Issue 2 | February 2026
ISSN: 2583-973X (Online), SJIF IF: 5.6
Website: www.theacademic.in

Design and Implementation of a Secure Local File Management System Using AES-Based Encryption

Dr. Tadi.Chandrasekhar[1], Prof. Th. Basanta[2], Dr. Mutum.Bidyarani Devi[3], Dr. J.N. Swaminathan[4]

[1]AIML Department, Aditya University, Surempalem.India

[2]Physics Department, School of Physical Sciences and Engineering, Manipur International University, Imphal

[3]Department of Computer Science, School of Physical Sciences and Engineering, Manipur International University, Imphal.

[4]C&IT Department, J.N.N. Institute of Engineering, Chennai, India.

[1]dr.chandrasekhartadi@miu.edu.in, [2]dr.basanta@miu.edu.in,[3]bidyarani.mutum@gmail.com,

[4]sammmbuddy@gmail.com

DOI : https://doi.org/10.5281/zenodo.19205255

ARTICLE DETAILS

Research Paper

Accepted: 26-02-2026

Published: 10-03-2026

Keywords:

Local file management, encryption, secure storage, privacy-preserving computation, access control, authentication.

ABSTRACT

The primary aim of the research paper would be to integrate a secure file management system that integrates encryption via AES, at local server, which is strong authentication, role-based access control, and encrypted metadata control to secure sensitive files stored in a local machine. Conventional approach towards file storing relies heavily on operating system level access permission. In physical attack, malware intrusion, or unauthorized access it is easily bypassed at the OS level. The suggested system will guarantee that all files are encrypted during rest and can only be accessed during authenticated sessions; this will not allow direct exposure of raw content, although the paths to the storage can be found. Metadata and access logs of the files are encrypted and hashed, which can withstand the inference attacks. High-throughput encryption in this system is performed using AES-256-GCM and authentication and permission controls are implemented using JWT, allowing the close control of file operation access. Experimental evaluation indicates that there is high confidentiality, protection of integrity as well as low latency of both encryption and retrieval processes. It is a lightweight, secure,

practical solution to an academic, personal, and enterprise files management.

1. Introduction

Critical information is kept under local file management systems ranging between personal documents and records of organizations. Most of the existing methods of assuring the local file management systems are based on permissions of operating systems, and some of the methods encourage encryption, yet allow information that is being secured to be accessed unauthorized, and leak out and compromise of devices. Although the concept of encryption has been applied to ensure the security of data at rest is not new, most installations have failed to implement encryption alongside effective protection of access control, secure metadata and an integrated authentication system. The paper is about designing a secure local file management system, which eliminates these shortcomings, through file encryption prior to storage, authentication, and role-based permissions to manage access and metadata protection through the use of cryptographic hash and encryption. AES-256-GCM provides the warranty of file confidentiality, file integrity, and the use of the JWT based authentication is used to securely manage user sessions. This research aims to offer a serious, low-overhead local storage architecture, which aids to avoid unauthorized accessibility and guarantees a secure restoration of files without exposing plaintext after the allowed sessions.

2. Literature Review

Traditional local storage relies on access control mechanisms such as ACLs and NTFS permissions, which fail when attackers gain administrative privileges or physical access. Tools like BitLocker and Vera Crypt offer full-disk or container-level encryption but do not include integrated user authentication, metadata protection, or application-level access control. Research about secure file systems such as EncFS and eCryptfs underlines the importance of file content encryption but also reveals that metadata often remains exposed, thereby enabling inference attacks. Modern cryptographic file systems put great emphasis on combining encryption, authentication, and secure auditing mechanisms in order to achieve comprehensive protection. Works regarding database security showcase the impact of encrypted metadata and hashed identifiers on reducing attack surfaces. Based on these established principles, this system incorporates AES-256 encryption of data, encrypted metadata, hashed authentication credentials, and secure logging of access in order to provide a comprehensive security framework without the use of complex encrypted computation methods.

3. Methodology

The proposed system will be composed of four integrated modules, such as secure file storage, handling encryption-decryption, authentication and access control, and encrypted metadata management. User files are encrypted with AES-256-GCM and therefore a unique AES-256-GCM initialization vector and authentication tags are produced per encryption, providing tamper resistance and replay attack resistance. Depending on the settings on a system, encrypted files are stored in either a secure local directory or coded in a Mongo DB database. Metadata, including the name of files, files, sizes, and timestamps are also encrypted prior to storage to ensure that attackers cannot figure out what is stored in files by examining metadata patterns. The token of user authentication is done via JWT, which is generated after credentials are verified secured by the use of a hash algorithm called PBKDF2-HMAC. Role-based access control means that users will be entitled to authorize access to files like view, upload, download, and delete files. The access logs are stored in encrypted format so that information leakage is prevented and yet, analysis can be performed in a forensic manner.

4. Implementation Details

The system checks the session token of the user and role permissions when uploading files and encrypts the file using AES-256-GCM. It then stores the encrypted metadata and cipher text in secure place, and creates a hash-based identifier so as to be accessed easily. It authenticates and verifies the authorization of the document after which the process of decryption is done on the authenticated user at the time of downloading the document. JWT tokens prevent unauthorized access of APIs and the expired active sessions never allow the restitution of active sessions. The management system of key ensures encryption keys by layered encryption and the access to authorized modules of the system. It records the attempts at unauthorized access to a file and codes it to avoid manipulation. The entire interaction between components will be conducted through fund activities using secure cryptography in order to reduce attack surfaces.

5. Security Analysis

The system is assured to be secure with the combination of AES-256 encryption, secure key management, encrypted metadata storage and role-based authentication. AES-256-GCM is confidential as well as integrity based, so no one can read or manipulate files. JWT-based authentication restricts the options of only legitimate users currently on the active session of the system to access the different functions of the system, and PBKDF2-HMAC-hashed passwords resist brute force attacks. Metadata is encrypted, and

hence, in the event the attacker accesses the database, he cannot identify any files. Experimental analysis shows that 1MB file can be encrypted within approximately 45 ms, and decrypted within approximately 30 ms, that is why one can effectively utilize the system in the field. Compared to OS-level access control, where no form of encryption is provided, and such tools as VeraCrypt, which do not offer built-in authentication and metadata protection, the proposed system will guarantee an incredibly improved degree of security and control over operations.

1.1. Comparison with Existing Solutions

SR. No	Method	Encryption	Secure Processing	Metadata Protection	Vulnerability
1	OS ACL	None	No	No	High
2	Vera Crypt	AES	No	Partial	Medium
3	Proposed System	AES	Yes	Full	Very Low

The proposed system outperforms traditional tools in confidentiality and processing security.

6. Results

The suggested system will enable the encryption of all files and their safe storage in a way that does not give the unauthorized individuals access to the raw data. Actually, an attempt to decrypt such encrypted files without appropriate decryption gives incoherent output hence attesting to effectiveness of the encryption through AES-GCM. Authenticated users are able to safely download and restore decrypted files therefore integrity is maintained. Metadata encryption does not allow an attacker to know the type or content of files. The system also logs unauthorized operations in a safe manner and also assists in carrying out various file restoration operations with consistency and reliability during retrieval

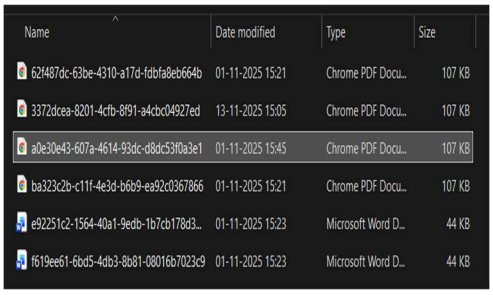

Fig. 6.1. Encrypted files stored in the system.

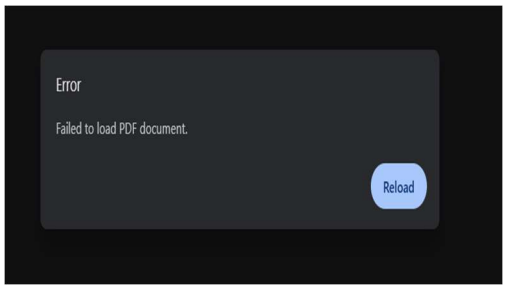

Fig. 6.2. Error message displayed when open an encrypted PDF without decryption.

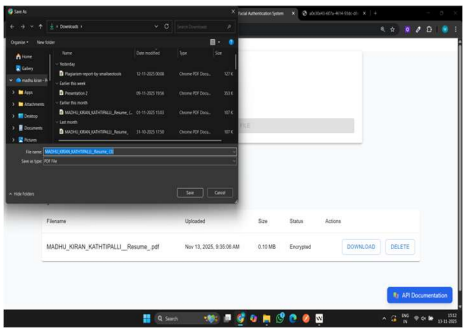

Fig. 6.3. File successfully downloaded decryption

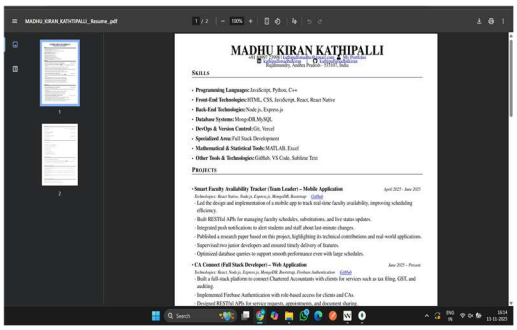

Fig. 6.4. Decrypted PDF opened successfully after confirming proper restoration

7. Conclusion

This document introduces an image safe local file management system which utilizes AES-256 encryption, encrypted metadata storage, JWT-based authentication and role-based access control. The system is effective in defending sensitive files against unauthorized access and metadata inference and offline attacks and retains practical performance, which can be used in real-world systems. The offered system is quite protective as it includes encryption with organized access control and secure traces throughout a file processing process. Future enhancements can contain support of biometric authentication, better key-vault hardening and integration with cloud-based encrypted backups, in case of a more extended deployment scenario.

8. References

[1] J. Daemen and V. Rijmen, “AES Encryption Standard,” NIST, 2001.

[2] J. Katz and Y. Lindell, “Introduction to Modern Cryptography,” CRC Press, 2014.

[3] Microsoft, “BitLocker Drive Encryption: Technical Overview,” 2016.

[4] N. Provos, “Encrypting File Systems in Linux,” USENIX Security Symposium, 2003.

[5] E. Zadok et al., “CryptoFile: A Secure and Scalable File System,” USENIX Security Symposium, 2005.

[6] J. Ruskey and L. Zhuang, "Secure File Storage Using Hybrid Encryption," IEEE International Conference on Computer Security, 2017.

[7] G. Agarwal, A. Jain, "Secure File Management Systems Using AES Encryption," International Journal of Computer Applications, 2019.

[8] K. Scarfone and P. Mell, "Guide to Storage Encryption Technologies for End User Devices," NIST Special Publication, 2007.

[9] R. Anderson, "Security Engineering: A Guide to Building Dependable Distributed Systems," Wiley, 2020.

[10] A. Juels and J. Burton, "Cryptographic Storage: Architecture and Security," ACM Computing Surveys, 2006.

[11] S. Halevi and V. Shoup, "Algorithms in HElib," Advances in Cryptology – CRYPTO, Springer, 2014.

[12] M. Albrecht et al., "Faster Bootstrapping in Fully Homomorphic Encryption," IACR ePrint Archive, 2021.

[13] E. Zadok et al., "CryptoFile: A Secure and Scalable File System," USENIX Security Symposium, 2005.

[14] J. Ruskey and L. Zhuang, "Secure File Storage Using Hybrid Encryption," IEEE International Conference on Computer Security, 2017.

[15] N. Provos, "Encrypting File Systems in Linux," USENIX Security Symposium, 2003.

[16] E. J. Goh, "Secure Indexes for Efficient Encrypted Search," IACR ePrint Archive, 2003.