Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 7s (2025)

Secure File Sharing Platform Using Public Cloud

Shifa Anjum[1], Mohammadi Begum[2] ,Sajida Nazneen[3],Eramma[4], Nagamani S.A[5]

[1,2,3,4]Assistant Professor, Krupanidhi College of Commerce and Management.

[5]Senior Technical lead QA, AON Consulting Pvt. Ltd, Bangalore

Article History:

Received:10-06-2025

Revised:27-07-2025

Accepted:10-08-2025

Abstract:
This research project proposes a secure file sharing platform using public cloud storage of AWS S3 with end-to-end encryption. This implementation uses a hybrid architecture where Node.js is used as the backend with the AES-256 symmetric encryption for files and RSA public key encryption for key management using OpenSSL. The two-factor authentication with the access token and OTPs over SMS is used for decryption and download. Uploaders have the overall rights for revocation mechanism and administrators read only audit logs and system status with the Role Based Access Control. Performance Testing with Apache JMeter stress testing provided a consistent throughput and response rates with encryption overhead. The error rates were primarily because of intentional security features like rate limiting and access validation and not because of a system inefficiency. OWASP ZAP Security Scanning provided excellent protection and blocked 100% of over 1,800 simulated attacks like SQL injection, Cross site scripting, and Command injection attempts. The outcome exhibits excellent scalability, performance, and the balance of security. The platform is well suited to sensitive environments such as business, finance, and healthcare where secure auditable data exchange is critical.

Keywords: Secure File Sharing, AES-256 Encryption, RSA Public Key Cryptography, Two-Factor Authentication, AWS S3, Access Control.

1. INTRODUCTION

Secure file sharing is most essential in current times the companies and the individuals freely share confidential files such as contracts certificates medical reports or legal papers. But traditional file sharing solution hardly provides confidentiality Integrity and Access controller that is required for secure file transfer which lacks features such as secure encryption access tracking file expiration and role-based authorization which is leading to greater risk of data leakage and violations.

To mitigate such attacks, we present a Secure File Sharing Platform in which users can upload, encrypt, share, and manage their files with the robust security. This system is constructed based on a hybrid approach of encryption model integrating AES (to encrypt the file data) and RSA (to encrypt keys AES) so that the contents of the files cannot be accessed by cloud storage providers. AWS S3 is employed to store all the encrypted files, Supabase to manage users and authentication and JWT tokens to handle the file expiry.

Other security features are also obtained through SMS based OTP authentication, file expiry timers, one time download enforcement, token revocation features. Administrators are also offered with a robust dashboard with audit logs, analytics, and health monitoring of entire system features.

This paper proposes and implements the following features for secure file sharing:

- Hybrid cryptographic file sharing architecture with the fine-grained expiration control.
- A secure public cloud used for time sensitive file sharing.
- Real time logging and analytics for traceability and audit.
- OTP protected download with expiration-based access control.
- One time download and token revocation capabilities.
- Role-based access and secure routing using Supabase Auth.

2. LITERATURE REVIEW

Secure file sharing in cloud has been studied in different architectures, encryption schemes, and access control models. Although these solutions have pushed the field, they are mostly constrained by usability loopholes, excessive overhead of computation, or poor real time access control integration. But most commercial platforms like Google Drive, Dropbox and WeTransfer are more concerned about the usability than a strong security enforcement, and most of them are not concerned about usability.

Token based access control has been investigated as an extension of static ACLs to support more dynamic policy enforcements [1]. Attribute based policy and hybrid models of control provide greater traceability and flexibility but typically omitted token expiration or OTP mechanisms [2]. Role based encryption framework enhance the Internal access structure but failed to enforce a multi factor authentication and revocation-based controls [3]. Incorporating a blockchain system with a robust term encryption algorithm like XChaCha20 and SHAKE will provide a tamper proof algorithm logging and immutability [4]. They are however usually affected with excessive latency along with poor real time adaptability. Attribute based Encryption is still studied for a fine-grained accessibility but it is challenging in terms of key distribution, scalability, and re encryption [5]. File fragment and a multi-layer storage architecture that have shown the enhancement in redundancy and a partial retrieval efficiency but lack secure download support authentication, link expiration [6][7]. Multi-tenant clouds have incorporated layered security architecture with the focus on policy enforcement but lack real time dynamic access and auditability [8].

Backup oriented address systems integrity and availability through encrypted redundancy but not a session-based accessor and misuse detection [9]. Techniques like segment-based deduplication or chunking are optimized through but the reducer latency regardless of file sharing policies [10]. Security service that has been included the cloud threats and the countermeasures recommending next generation cryptographic techniques such as homomorphic encryption and zero trust architecture [11]. These are the few implementations information that include recoverable real time access tokens.

The reviewed literature consistently identifies gaps in:

- Short lived, revocable token-based file access.
- OTP based download verification.
- Real time audit logging with the export capabilities.
- Integrated with public cloud APIs like AWS S3.
- End to end encryption combined with the role-based access.
- Cloud native implementation that integrates hybrid encryption (AES + RSA).
-

3. METHODOLOGY

The methodology focuses on the secure design, encryption logic, access control monitoring, and file lifecycle management in a public cloud-based file sharing system. The platform is engineered to provide an end-to-end data confidentiality, fine grained token-based access, real time auditing, and OTP based verification for secure download. The following components describe the design methodology:

3.1 System Design and Planning

The initial phase involves gathering both the functional and non-functional requirements for the secure file sharing platform. Security requirements included a strong encryption standard, multi factor authentication and a stricter access control mechanism. Non-functional requirements focused on performance, scalability and resilience against common cyber threats based on the requirements, the technology stack was finalized, incorporating Next.js for frontend, Node.js for backend, Supabase for database services, and AWS S3 for cloud storage. The encryption approach adopted a hybrid model combining AES-256 for data encryption and RSA for key encryption.

3.2 System Architecture

The secure file sharing platform has been architected as a multi layered, security first system to ensure that confidentiality, integrity, and controller access to shared data. The overall system designer focuses on scalability, fault tolerance, and compliance with the best security practices, integrating cloud storage, encryption, and role-based access control (RBAC).

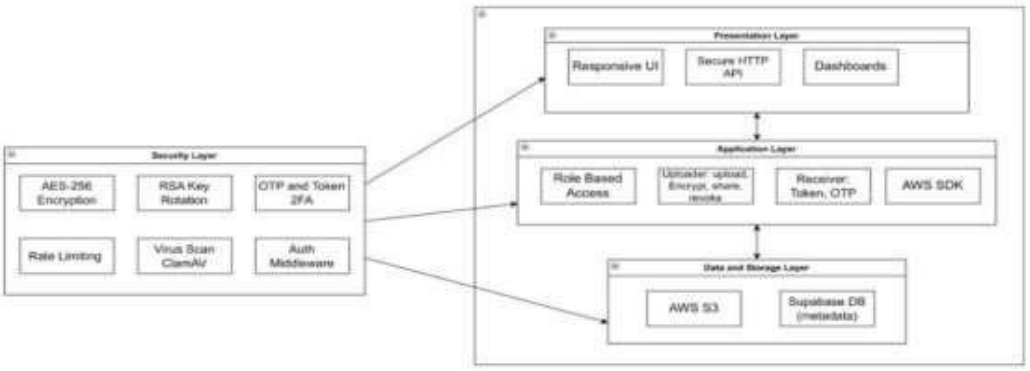

Figure 1: System Architecture Diagram

Figure 1 illustrates the core architectural layers of the system. Here the how the user interface interacts with the APIs for communication is take care by presentation layer. The complete system logic. The integration between the components is enforced through APIs and the complete logic from uploading and downloading. The files and their details are stored in the data and storage layer. The security layer makes sure that everything stays safe through encrypting and decrypting the files.

1. Presentation Layer which is developed using a Next.js it represents the user interface where users can upload or download files. What user sees depends on the user role and the application safely communicates to the backend using a secure HTTP based APIs.

2. Application Layer where all main work happens which is developed using Node.js and Express.js to handle all the logics. This layer does not allow any unauthorized access to any page this was done using Role Based Access Control (RBAC).

a. Uploader: can upload, encrypt, share, and revoke files.

b. Receiver can access files only with both access token and OTP.

c. Admin can view dashboards, audit logs, and system health status.

3. Data Storage Layer: This layer is used to store all the files in the cloud storage where private access policy and the pre signed links are used to prevent any unauthorized access. Supabase which is employed as the database to store information such as sender mail, receiver mail, expiry time of the file and the download history.

4. Security Layer: At this layer AES-256 symmetric encryption is used to securely encrypt the files before uploading into the cloud. The randomly generated keys of AES are also encrypted using RSA public key encryption where the key pairs are obtained using OpenSSL. The rate limiting is used to restrict brute force and DDOS attacks. The file is scanned for virus using ClamAV to ensure that the file does not contain any malware or virus all the endpoints which are sensitive are secured against a strong authentication middleware and checks offer role verification.

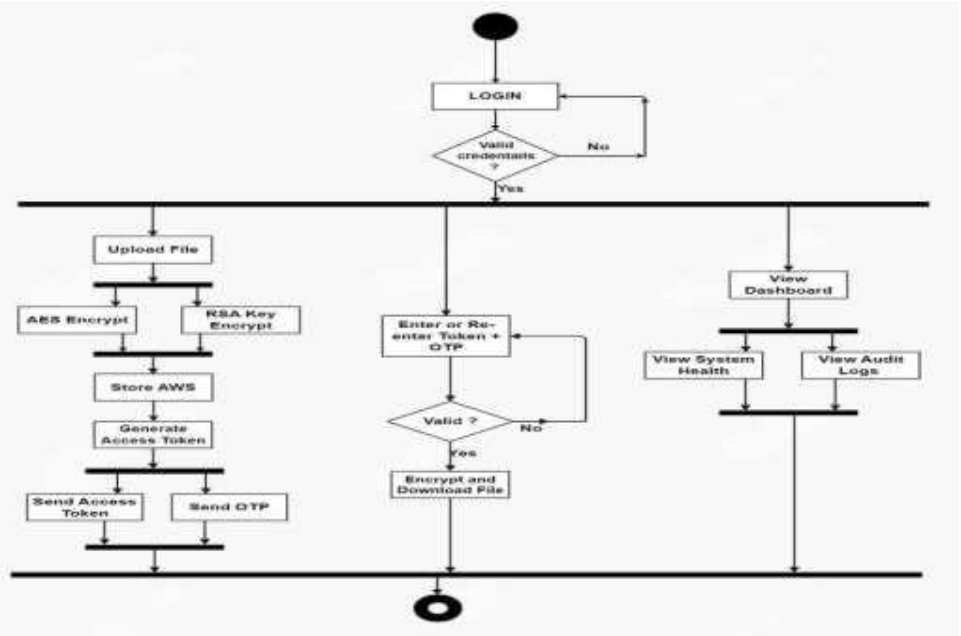

Figure 2: Complete workflow of the system using activity diagram

Figure 2 illustrating the complete workflow for user, admin, and receiver operations, including authentication, role-based access, secure file upload, token, and OTP distribution, and monitored file retrieval.

System Workflow

1. Authentication and Role verification: The user can access the system with appropriate login credentials when user enters the valid credentials then the credentials are shipped at the securely hashed password. That is encrypted using bcrypt cryptography package. After that authentication is successful the database will provide the access token and the roles information of the user. The verification of the role is validated at the route level of the page and at the controller logic of API to ensure that no unauthorized access attempts are not passed through.

2. Upload Flow: During the uploader process each file is checked for virus by ClamAV in prior to uploading form file to the cloud. ClamAV is an open-source antivirus engineer which is used to detect potential malware and virus. Once confirmed that the file is secure then the file is encrypted with the help of AES-256 encryption in which AES key itself is encrypted with the help of RSA public key encryption. The encrypted file is then upload to the AWS S3 private bucket using Cloud upload process. Upon successful upload the system then will create a unique access token. This token is not embedded directly into download link instead; the platform uses a Token Sent Separately approach. The system sends an access token via email and a onetime password (OTP) via SMS. The uploader still can revoke the file before the file expiration or download.

3. Receiver Flow: The receiver can download the file by clicking on the secure link that has been sent through mail and then receiver must input both access token and the OTP to download the file. The system verifies the inputs upon successful verification file is decrypted and downloaded from AWS S3 bucket.

4. Admin Flow: The admin can view the dashboard, audit logs, and check system health but admin can remove or access the files where the access of the file remains with the uploader.

4. RESULTS

The implemented secure file sharing platform has given the satisfied results, offering a seamless and a secure process for the file upload, storage, and controlled access for the uploaded file.

The screenshots of the implemented application are shown below (Figure 3 to 11).

Figure 3 to 6 shows the UI for Uploader Functionality. Uploader could login, select files, and initiate an upload process that automatically performs a virus scan using ClamAV an open-source antivirus engine that is used to detect the virus in the files and it is Docker containerized, encrypting the file using AES- 256, and store it in the AWS S3 buckets. Upon completion the system generates expiry download token and an OTP, both sent separately via email and SMS respectively. Uploaders will still retain the ability to revoke the access before expiry, ensuring complete control over the shared files.

Figure 3: uploader interface for selecting and uploading files.

Figure 3 shows the fields required for the uploader to input for the file upload such as phone number, email of the receiver and the file to be shared.

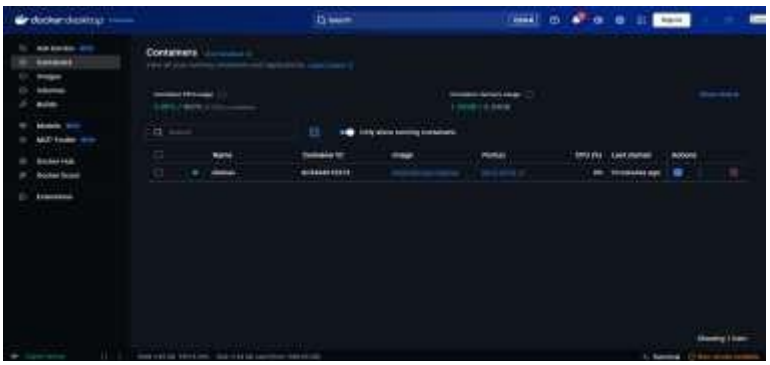

Figure 4: Screenshot of ClamAV virus scan process inside Docker.

Figure 4 shows us that how the ClamAV is containerized and used to scan virus and malware in the files using Docker.

Figure 5: AWS S3 bucket view showing the encrypted files stored securely.

Figure 5 shows how the file shared by the uploader is stored in the AWS S3 bucket using pre signed URL.

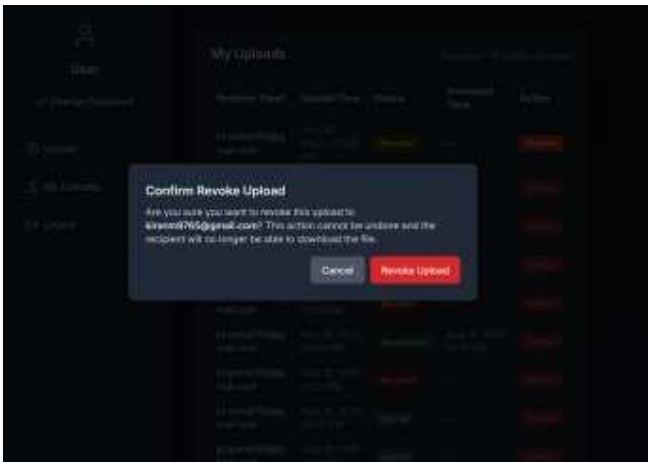

Figure 6: Revocation option available to uploader for shared files.

Figure 6 illustrates on the uploader can revoke the file access before the expiration and the download by the receiver.

Figure 7 to 9 shows the UI for Receiver Functionality. Receivers access the file download page via secure link but they are required to enter both access token and OTP for the verification. This two-step authentication efficiently prevents the unauthorized access. Successful validation triggers file decryption on the server side before download.

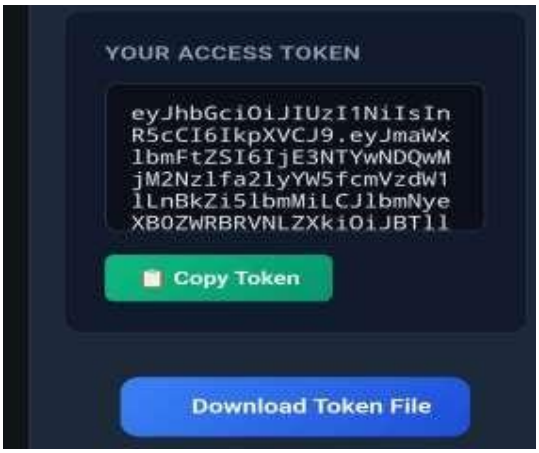

Figure 7: Email containing the access token and the secure download link.

Figure 7 shows on how the access token and the secure link for the download is shared to the receiver via mail.

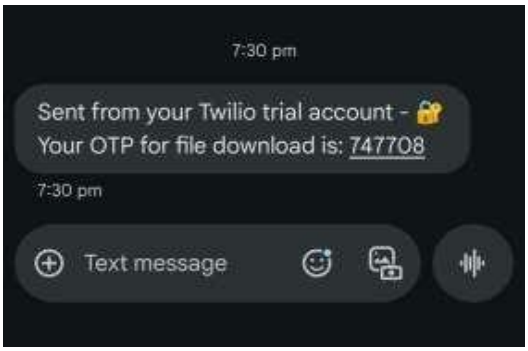

Figure 8: Example of SMS containing the OTP.

Figure 8 shows the example of how the OTP is sent through the SMS to the receiver for 2 factor authentication.

Figure 9: Receiver input screen for access token and OTP verification.

The figure 9 shows the fields required by the receiver to input after the secure link is opened to download the file.

Figure 10 to 11 shows the UI for Admin Functionality. Admin has access to monitoring focused dashboard displaying system health, audit logs, and usage statistics. However, they did not have the authority to revoke the file access this remains with the uploader who as the exclusive privilege on the files uploaded.

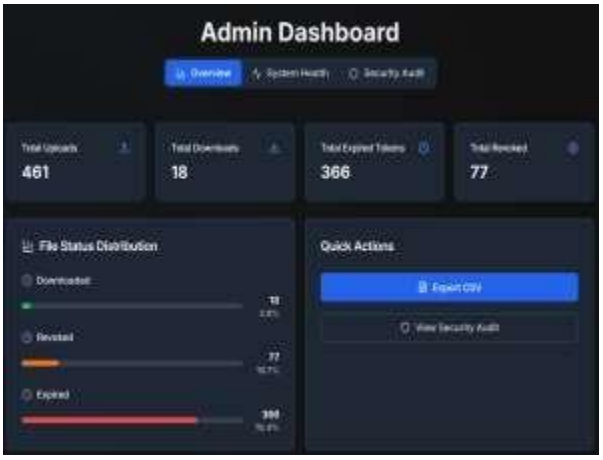

Figure 10: Admin dashboard displaying audit logs and total statistics.

Figure 10 shows the complete details of the system such as total uploads, total downloads, total expired tokens, total revoked.

Figure 11: Admin dashboard displaying the system health.

Figure 11 shows the complete health monitoring of the system as all the components are connected and their response times as well.

System Reliability: The architecture demonstrates a proper consistent, reliable system with no downtime recorded during a functional testing. All integrated components frontend, backend, database, and cloud storage AWS S3 operate cohesively to deliver expected functionality without critical errors.

Results Discussion

The Security and Performance Analysis of the Secure File Sharing Platform was conducted using two industry standard tools Apache JMeter for performance and load testing and OWASP ZAP for vulnerability and penetration testing. This combined approach ensured that the system was evaluated not only for speed and scalability under load but also for a resilience against common and advanced security threats.

Table 1: Testing results of OWASP ZAP

Attack Type	Attempts	Blocked	Success Rate of Attacks
SQL Injection	285	285	0%
XSS (Cross-site Script)	240	240	0%
Command Injection	195	195	0%
Template Injection	165	165	0%
File Inclusion	142	142	0%
Directory Traversal	118	118	0%
Information Disclosure	95	95	0%
Header Injection	85	85	0%
CSRF	72	72	0%
Miscellaneous	475	475	0%
Total	1,872	1,872	0%

Table 1 shows the testing results using OWASP ZAP. The OWASP ZAP spider scan was performed to automatically all the active endpoints. Then active Scan targeted all critical endpoints, including authentication routes, file upload or download APIs, user management APIs, and admin panel. A total of 1,872 simulated attack requests were executed across multiple vectors including SQL injection, Cross Site Scripting, command injection, template injection, file inclusion, directory traversal, information disclosure, header injection, CSRF testing, miscellaneous attacks. The results have shown that all the 1872 attacks are blocked

successfully by the end points. Hence, the success rate of attacks is 0%. In other terms, the success rate of the end points is 100%.

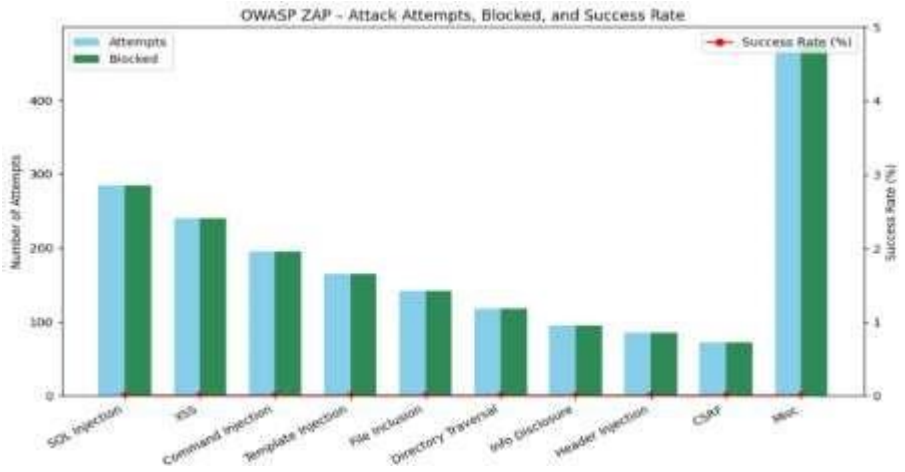

Figure 12: Attack Type vs. Attempt Count vs. Success Rate

Figure 12 illustrates the number of attempts for each attack type alongside the number successfully blocked, with a success rate overlay. The results show a 0% success rate across all categories, demonstrating robust security controls.

All the attacks were successfully blocked by a combination of input validation, rate limiting, and secure code practices, resulting in 0% success rate for every attack type

Table 2: Response Time of End Points during Attacks

Response Time Bracket	Requests	Percentage
0–1 ms	245	13.1%
1–2 ms	428	22.9%
2–3 ms	385	20.6%
3–5 ms	312	16.7%
5–10 ms	285	15.2%
10–15 ms	142	7.6%
15–30 ms	75	4.0%
Total	**1,872**	**100%**

Table 2 shows the response time of each end points during the simulated attacks. 56.6% of the requests were processed within 0-3ms which indicates that the system maintained low latency even under heavy traffic. Around 31.9% requests were handled within the 3-10 ms. While only the small part of 11.6% went beyond the 10 ms.

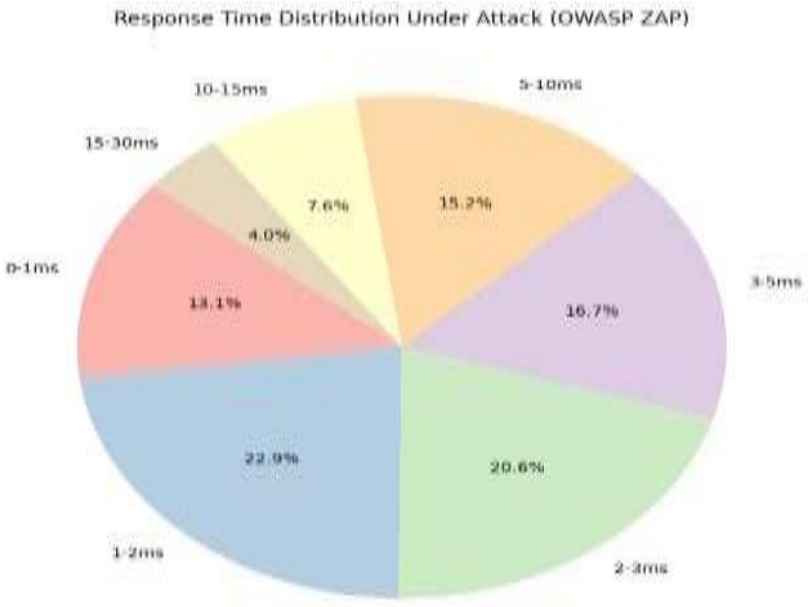

Figure 13: Response Time Distribution Under Attack

Figure 13 presents the percentage of requests served within different response time brackets during the OWASP ZAP active scan, highlighting that the majority were processed within 0–3ms.

Key security observation:

- Rate limiting returned the status 429 which indicates Too Many Requests for a high frequency attempt, mitigating brute force and automated attacks.
- No SQL injection vulnerabilities were successfully
- CORS policy and session management were correctly implemented.
- The download endpoint had a 0% success rate for all the attack types. Performance testing was conducted under baseline and stress scenarios. **Baseline load tests:**

Baseline testing is the process of running a set of tests to establish a benchmark for the comparison

- Login: 200ms average response time
- File Upload: 1.5 seconds average (due to encryption, AWS S3 upload, token generation and OTP dispatch)
- File Download: 30ms average (successful authorized download)

Stress Test:

Stress Testing is a type of software testing where the system is tested under extreme conditions to see how it behaves when it goes beyond normal limits.

- upload times increased to 2.88 seconds due to additional SMTP OTP sending overhead and encryption steps.
- Download endpoint showed 100% error rate for unauthorized requests confirming security enforcement
- Login and dashboard maintain the sub 300ms response despite the load.

Table 3: Baseline vs Stress Test per Endpoint

Endpoint	Baseline (ms)	Stress Test (ms)
/api/server-conn	22	35
/api/auth/login	200	271
/api/upload	1500	2881
/api/download	30	35

Table 3: compares the response time of end points under baseline and the stress testing. The test resulted as the end points /api/server and /api/auth/login showed a moderate increase in the latency under stress test. The endpoint /api/upload had more increase in the latency due to the encryption overhead, AWS S3 cloud upload, and repeated OTP dispatches. While the /api/download end point had a consistent low response time which shows the efficient handling.

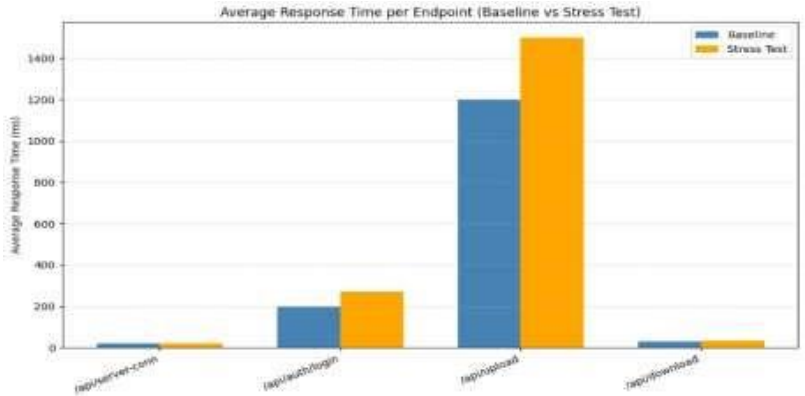

Figure 14: Average Response Time per Endpoint (Baseline vs. Stress Test)

Figure 14 illustrates the comparative analysis of average response times for key endpoints under baseline load and stress conditions, demonstrating predictable performance despite increased latency.

Table 4: presents the error rates per endpoint during the stress testing.

Endpoint	Error %
/api/server-conn	14.1%
/api/auth/login	6%
/api/upload	8–12%
/api/download	100%

The /api/server-conn reported 14.1% error rate due to the intentional rate limiting on the repeated requests. The api/auth/login endpoint showed a modest 6% error rate which is modest. The endpoint /api/upload experienced 8-12% error rate, which is due to the encryption overhead, AWS S3 cloud upload. While

/api/download endpoint recorded 100% error rate for unauthorized requests where the system enforces strict token and OTP validation. This confirms that error rates were not due to the backend inefficiency.

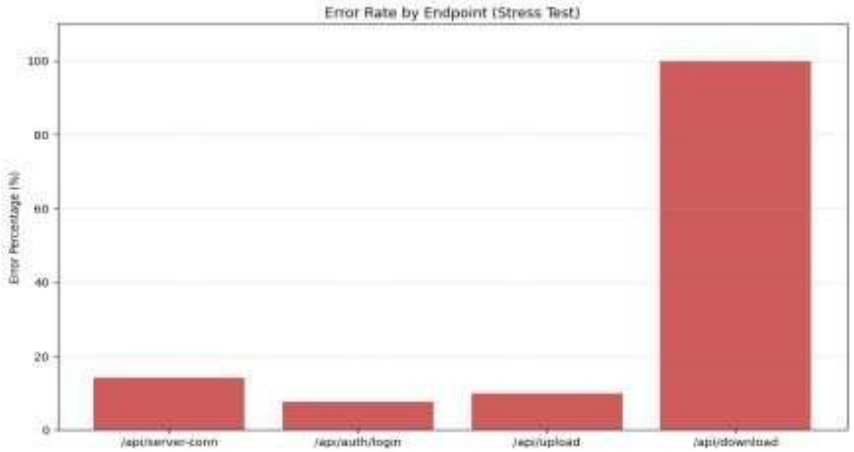

Figure 15: Error rates by endpoints during the stress testing.

Figure 15 illustrates varying error percentage across endpoints with 100% block rate on downloads for unauthorized access attempts which assures the strict enforcement of the security controls.

Performance Observation

1. Latency of encryption: it was due to AES-256 encryption, RSA wrapping of keys, and AWS S3 upload but the latency was within the acceptable range for the secure file transfer.

2. Rate limiting was enforced due to security over throughput, sometimes refusing a successive quick upload from the same client.

3. Server Stability: Even with the encryption and security checks there was no server crashes were encountered during the stress testing.

Interpretation of Results

The outcome of performance and security testing provide information that the secure file sharing platform is up to the industry expectation in terms of data security and system robustness. OWASP ZAP Scan provided the security score with zero successful attacks out of 1,872 attempts. The 100% defense rate proves that security features especially input validation, rate limiting, and the role-based access control are very effective.

Comparison with Expected Benchmarks

Security performance has met the industry levels whereby zero exploit success in all categories of vulnerabilities is the target. Most commercial file share solutions are unable to have a full protection in automated scanning the current systems score exceed those average performance wise, normally secure file uploads with the encryption and the cloud storage within 2 to 4 second range during stress the measured average of 2.8 seconds fall well within the benchmark Security versus performance and trade offs The architecture prioritizes data confidentiality and controlled access, which naturally introduces additional processing steps virus scanning, AES-256 Encryption, RSA Encryption, Token generation, OTP Dispatch. While these steps marginally increase response times, they significantly reduce their risk of unauthorized data access. The rate limiting which helps to prevent brute force and DDos attacks, also slows high frequency requests in a stress scenario but ensures system stability and up time.

5. CONCLUSION

This project successfully designed and implemented a Secure File Sharing Platform with enterprise-grade security features, robust performance, and fine-grained access control. AES-256 encryption, RSA key wrapping, Multi factor authentication, and AWS S3 cloud storage ensured the confidentiality and restricted unauthorized access of data. The system architecture is stronger and highly secure under most of the rigorous testing.

Limitations

1. SMTP authentication overhead, repeated SMTP logins for Token delivery during a stress testing added latency and contributed to slightly higher error rate for upload.

2. Single region cloud storage current AWS S3 is region specific which may impact latency for users geographically distant from a chosen region.

3. No offline access mode is the platform strictly requires a network availability for an authentication token validation which may be limiting in a restricted connectivity environment.

Future work

- To implement an optimized pipeline Approach for AES-256 and RSA operations by parallelly running virus scanning, encryption, and cloud upload which can reduce the upload latency.
- To integrate advance key management measures such as HSTS, CSP, and hardware security modules.
- To deploy load balancing, auto scaling to optimize OTP or email handling to reduce delays under heavy load.

6. REFERENCES

1. MacLennan, J. & Zhang, J. (2024) Path-safe: enabling dynamic mandatory access controls using security tokens. Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON 2024), 15–18 July 2024.
2. Zubair, M., Sabzevari, M., Khatri, V., Tarkoma, S. & Hätönen, K. (2024) Access control for trusted data sharing. EURASIP Journal on Information Security, 2024(30).
3. Wang, S., Wang, X. & Zhang, Y. (2019) A secure cloud storage framework with access control based on blockchain. IEEE Access, 7, 112713–112725.
4. Morales-Sandoval, M., Cabello, M. H., Marin-Castro, H. M. & Compean, J. L. G. (2020) Attribute-based encryption approach for storage, sharing and retrieval of encrypted data in the cloud. IEEE Access, 8, 170101–170116.
5. Guo, C., Su, M. & Cui, F. (2023) Research on data storage security in cloud computing environment. Proceedings of the 4th International Conference on Information Science, Parallel and Distributed Systems (ISPDS 2023), 14–16 July 2023.
6. Suman, O. P., Saini, L. K. & Kumar, S. (2023) Cloud-based data protection and secure backup solutions: a comprehensive review of ensuring business continuity. Proceedings of the 3rd International Conference on Secure Cyber Computing and Communication (ICSCCC 2023), 26–28 May 2023.
7. Zhang, D., Deng, Y., Zhou, Y., Li, J., Zhu, W. & Min, G. (2023) MGRM: a multi-segment greedy rewriting method to alleviate data fragmentation in deduplication-based cloud backup systems. IEEE Transactions on Cloud Computing, 11, 2503–2516.
8. Eswari, R., Vamshi, A. & Sultan, M. S. (2023) An efficient data storage technique for user files in cloud. Proceedings of the International Conference on Quantum Technologies, Communications, Computing, Hardware and Embedded Systems Security (iQ-CCHESS 2023), 15–16 September 2023.
9. Santhoshkumar, S. P., Hariharasudhan, S., Prakash, S. P., Philip, J. M., Haritha, S. & Nalini, T. (2024) Security challenges and elucidations in cloud storage and file systems: an advanced investigation review. Proceedings of the 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS 2024), 12–13 December 2024.

10. Jayachitra, J., Arshad, H. & Muthukumaran, R. (2024) Cloud storage and secure file sharing using blockchain with XChaCha20 and SHAKE cryptography. Proceedings of the 16th IEEE International Conference on Computational Intelligence and Communication Networks (CICN 2024), 22–23 December 2024.
11. Kaushik, J., Jensen, J. C. & Vasanthi, R. (2025) Enhanced security through distributed fragmentation and encryption for cloud-based data storage. Proceedings of the International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI 2025), 28–29 March 2025.
12. Scopelliti, G., Baumann, C. & Mühlberg, J. T. (2024) Understanding trust relationships in cloud-based confidential computing. Proceedings of the IEEE European Symposium on Security and Privacy Workshops (EuroS&PW 2024), Vienna, Austria, 8–12 July 2024.
13. Raut, S., Patil, S. & Hsu, V. (2024) Efficient backup management in hybrid cloud deployment based on workload data classification. Proceedings of the IEEE International Conference for Women in Innovation, Technology & Entrepreneurship (ICWITE 2024), Bangalore, India, 16–17 February 2024.
14. Syed, A. A. M. (2024) Disaster recovery and data backup optimization: exploring next-gen storage and backup strategies in multi-cloud architectures. International Journal of Emerging Research in Engineering and Technology, 5(3), 104.