International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Intelligent Threat Detection and Secure File Exchange

MD ISMAIL KHAN[*1], PUCHAKAYALA KARTHIK[2], KOKKILIGADDA VENKATESH[3], SYED SAMEER[4], DEEKULLA ROHIT[5]

[1]Assistant Professor, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP,India
[2]Student, Department of CSE(CB),Bapatla Engineering College, Bapatla 522101,AP, India
[3]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India
[4]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India
[5]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India

***Abstract*—**Growing expansion of communication technologies has made secure file sharing needful for organizations, educational institutions and cloud-based service providers. Nevertheless, conventional file exchange systems are limited to encrypting the data without offering intelligent threat detection mechanisms. This is a limitation that make it easy for malware to spread, unauthorized access, leads to data leakage.

This works proposes an Intelligent Threat Detection and a Secure File Exchange System that can help in providing end-to-end security for file transfer activities. The ideal system incorporating secure authentication, real time malware scanning, encryption techniques, centralized logging and monitoring by administration is proposed. Signature-based and heuristic detection methods are used to identify malicious patterns in uploaded files before file transfer. The safe files saved and stored through encryption, while it blocks and report suspicious ones.

***Key Words*— Cybersecurity, Secure File Transfer, Malware Detection, Encryption, Threat Analysis, Activity Logging.**

II. INTRODUCTION

Modern organizations rely heavily on digital file sharing for communication, collaboration, and information exchange. However, as we become more reliant on Internet based File Transfer systems, the threat of cyberattack has also increased dramatically. Files with malware, unwarranted access, and data breaches have emerged as new trends in cyberspace threats.

Conventional file sharing services mainly depend on encryption methods for securing the data while in transmission. Encryption protects the payload of a network transfer, but there is always some risk of hostile data.This allows attackers to hide malicious scripts or malware inside encrypted files, which can evade conventional security layers.

These challenges necessitate the existence of intelligent systems that can detect and eliminate threats before the file transfer process even begins. Combining various components of malware detection, secure authentication, encryption and monitoring features into a cohesive system will help you achieve better cyber security.

Combining secure file transfer with intelligent threat analysis, the proposed Intelligent Threat Detection and Secure File Exchange System serves as a comprehensive solution. The allows or add data security, including malicious file sharing prevention in sensitized file transfer and user authenticity as well as online logging for any administration.

III. LITERATURE SURVEY

With the growth in the digital communication infrastructure, modern organizations have become heavily reliant upon secure file transfer mechanisms. Nowadays, these sensitive info are exchanged on the network. It increases exposure to cyber vulnerabilities like malware infiltration, ransomware infection, data hack, etc. Earlier studies only focused on securing data in transit with various encryption techniques. It was SSL (Secure Socket Layer) and TLS (Transport Layer Security) protocols that were commonly used to protect the communication channel from eavesdropping. While these protocols do encrypt the transmission of your data, they do not guarantee that any received files are not malicious. Consequently, encrypted communication alone cannot give full protection today's networked environments. [1].

Signature-based detection methods have been relied on by typical antivirus solutions for known malware detection. A file check system checks files for malware, ensuring the computer is safe for further tasks. In real time applications these techniques are generally effective, and computationally efficient, for detecting known attacks. Nevertheless, cyber attackers often modify the code of malware to create new variants that cannot be matched to any signatures. The constant evolution of malware has prompted the researcher to look for new detection methods to discover the unknown threats. [2].

To overcome the limitations of signature matching, heuristic and behavior-based analysis techniques were utilized. These methods look at file characteristics such as the structure the file uses, execution flow and unusual interactions with the operating system to determine whether it poses a risk. Heuristic scanning identifies new kinds of malware through suspicious patterns rather than defined signatures. However, misconfiguration of detection thresholds may result in a greater false positive rate and more alerts. As a result, hybrid detection models that combine signature-based and heuristic techniques have been proposed to achieve improved accuracy and system dependability. [3].

Research on malware detection has progressed with the adoption of AI and ML. Today's intelligent security systems make use of algorithms like Decision Trees, Support Vector Machines, and Neural Networks that analyze file attributes and notice abnormal behavioral trends. By using historical datasets and adapting to threat patterns, the models can be flexible compared to rule-based systems. Though advantageous, the machine learning methods need a vast quantity of training data and significant processing power, thereby limiting their applicability on light-weight file exchange mechanisms. [4].

In environments where files are shared, strong authentication mechanisms must also be implemented. According to the researchers, there is a need for strong identity verification techniques to curtail unauthorised access to these embedded systems. Authentication models that utilize tokens (especially JSON Web Tokens or JWT) are gaining much traction owing to their statelessness and ability to better control security. As a result, session management is secure and the potential for credentials misuse and session hijack is low. An upload or download can be initiated only by the client's or server's respective authorized user. [5].

Monitoring and logging solutions are also a popular area within cybersecurity research. Security Information and Event Management (SIEM) systems collect activity logs and user behavior analyses for unusual patterns which may indicate a threat. Administrators can find suspicious activities in real time with continuous monitoring. Suspicious log ins, data stealing and file uploads can be prevented. Using a logging framework will help create transparency and strengthen response activities by implementing logging. Many existing systems still do not have monitoring fairly comprehensive. [6].

Due to cloud computing, we've more secure options for data storage and file sharing. Experts have investigated a cloud framework that provides encryption, as well as distributed storage architecture and scalable resource consumption. These solutions boost accessibility and efficiency of the systems while ensuring data security. Cloud-based file sharing environments, however, bring about issues of data privacy, trust management and dependence on third parties. Continues to be an active area of research to ensure effective threat detection within cloud infrastructures. [7].

Cybersecurity in distributed applications has recently been studied through blockchain technology. Blockchain logging systems can develop unchangeable records of system events so any unauthorized modification or deletion of security logs is neutralized. It makes people more accountable and trustworthy regarding digital transactions. Nevertheless, employing blockchain incurs extra computing overheads and affects system performance while exchanging files in real time. [8].

IDS are extensively utilized for monitoring the traffic and identify probable happenings in the network. The packets of data cannot be decoded and sent through a channel. Research suggests that IDS combined with file scanning mechanism can greatly enhance the accuracy of intrusion detection by adding an additional layer. This comprehensive security plan allows organizations to detect malicious activity earlier and decreases the chance of malware traveling over networks. [9].

IV. EXISTING SYSTEM

Most of the current file sharing systems are limited to functionalities that support basic operations like uploading or downloading files, and encrypting your data during transfer. These systems maintain confidentiality with secure communication protocols, but unfortunately do not incorporate intelligent

strategies for detecting malware content pre-file transfer. Consequently, malware-infected files are shared between networks before being verified, significantly raising the readiness for cyberattacks, unwanted data exposure and system gained access. That would be easier than it is, however: Most traditional platforms are very reactive rather than proactive when it comes to security threats.

Moreover, most of the solutions available today come without centralized monitoring or detailed logging capabilities for administrator to audit activity. This becomes a hindrance when trying to monitor suspicious user activity or analyze file transfer in real time. Additionally, weak authentication controls and lack of integrated threat scoring also impacts system reliability. This means that companies that still utilize traditional file exchange systems are susceptible to data leakage, malware and unauthorized access, necessitating intelligent and secure data exchanging frameworks.

V. PROPOSED SYSTEM

Intelligent threat detection and secure file exchange system: A multi-layered solution that both addresses the limitation of traditional file sharing platforms. It validates documents for safety to make sure no harmful threats escape before files move over the network. The system combines authentication with threat detection, encryption, and monitoring to create a well-rounded security environment.

It starts with authenticated users to the platform. Upon logging in, users can upload files which are processed by the threat detection engine automatically. This module scans files with malware signatures and suspicious patterns, calculates a threat score, and classifies the file as either safe or malicious.

If it is deemed safe, the file will be encrypted and stored or sent securely to the end user. If the transfer is flagged as malicious, the system blocks it and creates alerts. A centralized logging system captures all activity, allowing administrators to monitor usage and identify suspicious behavior, ensuring overall system security and enabling transparency.

VI. SYSTEM ARCHITECTURE

1. Layer of user Interaction

At the user interaction layer, the user initiates communication with the system. Users of the application who upload and download files fall under this layer. Users use the system from a web browser that's simple to use and allows them to perform various operations.

The User Facilities layer allows users to log in, submit files, view reports, and so on. The system increases usability without sacrificing security controls through an intuitive interface.

2. Module for Client Contact

The client-side interface uses web technologies like HTML, CSS, and JavaScript. This module forwards the user inputs to the server. It ensures the necessary features such as user authentication forms, file uploads, and secure downloads.

At each stage, mechanisms are properly validated to ensure that incorrect or incomplete inputs are not forwarded to the backend. By avoiding unnecessary processing load, system efficiency improves.

3. Backend Server Layer

A back-end server as CPU of system and it is implementation of Python Flask. It handles communication between various modules and processes user requests from the client interface.

The layer handles the verification of authentication, processing of operation files, coordination of threat analysis, control of encryption and database operations. Thus, by concentrating the system logic on the backend server, this architecture has better control over security operations and maintenance.

4. Authentication section

The authentication module verifies user credentials at login and helps ensure that only the authorized user can access system resources. The manager validates passwords, creates sessions, and manages access permissions.

Passwords are stored in encrypted format using secure hashing techniques in the database. System security module ensures the system is protected against unauthorized intrusions and unwanted attacks.

5. It's about file handling module

The module for file handling assists in uploading the file and temporarily storing it and sending it to the target location. When a user uploads a file, the module receives the file and forwards the file to the threat detection engine to check for safety.

This will make sure that they enforce file size limits, format validation, and storage policies. This module helps the architecture maintain the integrity of data and prevents users from changing the file.

6. Threat Detection Software.

The proposed system has threat detection engine as one of its key security components. When you upload a file, it scans it using malware signature matching and heuristic scanning techniques to determine whether it is suspicious.

The engine assigns a threat score which reflects severity of trouble detected patterns If the file is malicious, the further processing is blocked and alert is created. If the file is safe, the file is forwarded to the encryption module. This proactive analysis significantly reduces the likelihood of malware spreading throughout the network.

7. Encryption Component

The encryption module secures safe files before they are stored or transferred. It utilizes cryptographic algorithms to transform files into encrypted forms inaccessible to non-permitted users.

Through the utilization of this module, the confidentiality of your data is protected. After encryption, these files are kept in the database for access by authorized users.

8. System for Keeping Records and Watching Over

All significant system activities including login attempts, file uploads, scan results, and download activities are captured in the logging module. The logs help to identify suspicious behaviour and inform where and how a system is used.

By checking these logs, administrators can monitor any such security incident and act accordingly in case of a threat. This module strengthens accountability and enables effective incident response.

9. Generation module scan reports

The report generator module creates detailed reports from the threat detection engine result. The reports have details like threat score, malicious patterns found, file status and time stamp.

The reports assist the administrator in understanding the security status of the system. They also allow the user to verify whether their files have been processed or blocked as a security concern.

10. Database Layer

SQLite has been used in the database layer for storing encrypted files, user credentials, activity logs, and scan reports. A secure storage mechanism guarantees that intelligent data is reliable and cannot be manipulated.

11. Administrative Control Layer

The administrative layer allows system administrators to monitor user activities, review security logs, and manage threat reports. This layer provides centralized control over system security and enables timely responses to suspicious events.

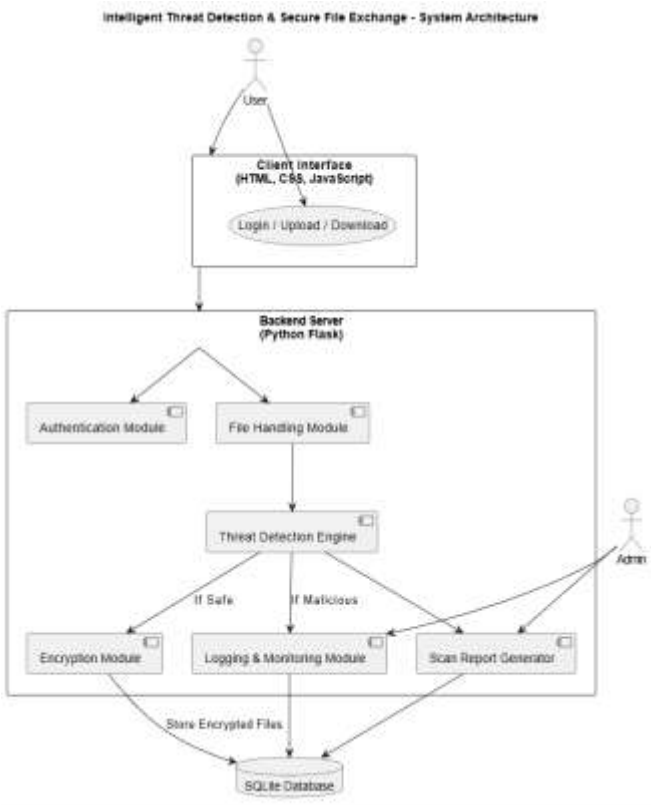

Figure 6.1 :System Architecture For Threat Detection

VII. METHODOLOGY

1. Method of Development

The system is developed using the iterative development method, where different modules were designed, implemented, and tested. The application is not constructed all at once but rather in steps based on the orderly application development process.

To create a secure access mechanism first the authentication, user management modules were developed. Once the restricted access of the system was established, the modules for files upload and threat detection were implemented. Later phases incorporated encryption, logging, and monitoring functionalities..

2. Design of System Architecture

In our system, the frontend interface and the backend server communicate through a secure client-server architecture. The web interface allows users to register, sign in, and drag and drop files into the system.The backend server examines requests, performs a threat analysis, encrypts files, and stores the system data. By decoupling user interface from backend processing, scalability and maintainability of the system can be enhanced. The design of each part of the architecture is such that they manage a particular task. This helps the system to effectively run even with simultaneous users.

3. User Login and Login Permissions

The authentication mechanisms must be strong. To begin the registration process, all users must register using valid credentials. Upon login, the system check the identity of the user before granting access.

Passwords are stored in an encrypted form to prevent access to sensitive data. To ensure authenticated users have secure connectivity with the system, the session management technique is implemented. Only users with valid login credentials can access the file exchange.

4. File Upload along with Threat Detection Process

When a user is authenticated, files can be uploaded via web application. Once a file is uploaded, the system automatically forwards it to the threat detection module for detection.

The file is examined by the threat detection engine using malware scanning techniques. It compares the signature patterns of the file to those of known malware in a database. If the system detects any malicious patterns, it will block the file and display a warning message.

5. Encrypting and Securing Files with Storage

After scanning, if the uploaded file is safe to encrypt, the process continues with encryption. Encryption keeps the file private while it is stored (in a computer) or transferred (to another).

The file that has been encrypted is then stored in the database. This extra layer of protection prevents unauthorized individuals from accessing the contents of stored files even if the storage system is compromised.

6. Keeping a record of activities

The logging module records all important activities in the system to make the system more transparent and security aware. The user logs include login attempts, file uploads, scan results and file downloads.

These logs can be reviewed by administrators to detect system malfunctions and disable abnormal behaviors. The logging system makes sure accountability is maintained and allows the administrator to investigate your security incidents, if there are any.

7. Testing and Verification

After implementing every module, the system was tested rigorously to ensure efficiency. For testing, we used some normal files, suspicious script files, and sample malware files.

The test checked if the system can block a user whose files transfer is infected with malicious content. The findings proved that the system detects any threats and does not slow down file processing.

VIII. IMPLEMENTATION

The implementation of the Intelligent Threat Detection and Secure File Exchange System took the form of a secure web-based system with a modular full-stack. The user interface is an environment that is designed to be simple and responsive. This part of the system was all created within HTML, CSS and JavaScript. Furthermore, the interface allows the user to register themselves, authenticate themselves, upload their files and view their reports. The backend functionality was done by using Python Flask framework to process requests, manage sessions, handling files operations and communication between disparate security modules. To ensure reliable data processing and prevent the impact of invalid submitted input data on system performance, client and server validations were adopted.

The system provides its users with a complete suite of security options. These protection options include real-

time threat detection, safe file encryption, and centralized logging of activities. Uploaded files are checked for viruses before storage or sharing. This prevents malicious files from being transferred in the first place. SQLite was chosen as the database to safely store user credentials, files, scan report, and system log. All modules were tested separately and assembled together to ensure easy functioning of the system. The final implementation provides a secure environment for regulated file exchanges. Furthermore, it enables better monitoring and threat awareness.

IX. RESULTS

1.User authentication and secure login

The login page is a secure access point that registered users can authenticate on with valid credentials. The system uses a secure log-in process to allow entry to the dashboard and file-exchange features. To prevent unauthorized access, we made use of techniques such as input validation and session management. The authentication tool's ability to deter unauthorized access certifies the effectiveness of the functionality.

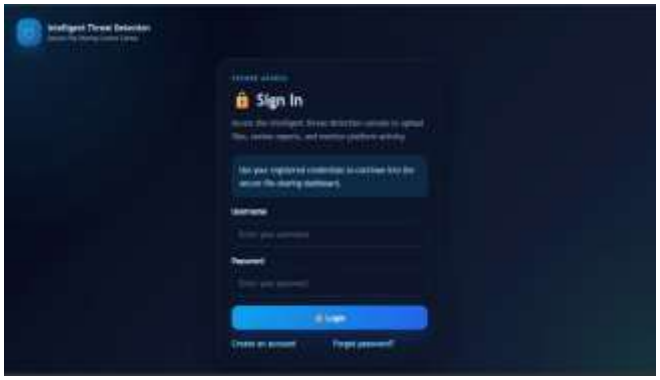

Figure 9.1 : Main Login Panel

2. Control Panel and User Area

Users are redirected to the security dashboard after a successful login. This is the controlling panel of the application. The status of the system, file upload and download options, and navigation for scan reports and security logs are shown in Figure 2 of the dashboard. The dashboard gives feedback on the system in real-time to enhance usability.

Figure 9.2 : User Dashboard Interface

3. File upload and Threat scanning

Users can upload files for threat analysis with the help of the file upload module. The system indicates the selecting and upload files for scanning as depicted in Figure 3. The threat detection engine scans the uploaded file automatically for signature matching and malware heuristics. With this scanning, malicious files get detected before they are saved or transferred to the user's device.

Figure 9.3 : Secure Files Uploading Panel

4. Report Generation of Threat Detection and Scanning

The interface of the scan report shows detailed report information of the file scan process. The system assigns threat score for suspicious behaviors (e.g. suspicious command execution or encoding injection) and is visible in the following diagram (Figure 4). Files with a high threat score are automatically blocked with a warning message. Through this function, it is ensured that malware cannot spread and cybersecurity will improve.

Figure 9.4 : File Scan Report Panel

5.Monitoring of Security Logging and Administration

The administrative security log panel maintains a record of major system activities including attempts to login, results of file upload and detected threats. Figure 5 demonstrates the log entries created by the system to log any event that happens in the system. By monitoring them all together, the administrator can detect unusual behavior and act upon a threat.

Figure 9.5 : Admin Security Logs Panel

X. CONCLUSION

The Intelligent Threat Detection and Secure File Exchange System offers a realistic and effective solution for increasing security in file transfer environments. By putting in place constant activity monitoring, real-time tracking mechanisms, encryption, and secure authentication, the system considerably decreases the risk of threat propagation and malware. Based on experimental results, the platform can analyze uploaded file contents, block malicious content and keep security logs for admin supervision. The use of a modular architecture and web implementation helps enhance usability, scalability, and maintainability of the system. All in all, the proposed system can strengthen cyber security practices through secure file exchange and active threat awareness in today's networked applications.

XI. REFERENCES

[1] W. Stallings, Cryptography and Network Security: Principles and Practice, 7th ed., Pearson Education, 2017.

[2] S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," IEEE Network, vol. 15, no. 4, pp. 24–31, 2000.

[3] J. O. Kephart and W. E. Arnold, "Automatic Detection of Computer Viruses," Proceedings of the IEEE Symposium on Security and Privacy, pp. 92–103, 1994.

[4] M. Schultz, E. Eskin, F. Zadok, and S. Stolfo, "Data Mining Methods for Detection of New Malicious Executables," IEEE Symposium on Security and Privacy, pp. 38–49, 2001.

[5] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," IETF RFC 7519, 2015.

[6] R. Behl and K. Behl, Cybersecurity and Cyberwar: What Everyone Needs to Know, Oxford University Press, 2016.

[7] R. Buyya, C. Vecchiola, and S. T. Selvi, Mastering Cloud Computing, McGraw Hill, 2013.

[8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[9] W. Lee and S. Stolfo, "A Framework for Constructing Intrusion Detection Systems," ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 227–261, 2000.

[10] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," NIST Special Publication 800-145, 2011.

[11] ClamAV Documentation, Cisco Systems Inc., Available: https://www.clamav.net

[12] Flask Web Framework Documentation, Pallets Projects, Available: https://flask.palletsprojects.com