Location Based File Integrity Monitoring & Preventing

Mr. P. Shanmukha kumar [1], G. Asha Jyothi[2], K. Jyothi[3], CH.Keerthi[4] CH. Assistant Professor[1]
Student/Research Scholar[2], Student/Research Scholar[3],Student/Research Scholar[4],
Department of Cyber Security Malla Reddy University, Hyderabad
Maisammaguda, Dulapally, Hyderabad, 500100, Telangana, India

Abstract — **File Integrity Monitoring (FIM) is a critical security mechanism used to detect unauthorized modifications, deletions, or additions to files. Traditional FIM systems focus on monitoring file changes but lack spatial awareness, which can be a vital context in identifying and mitigating potential security threats. The effectiveness of an integrity checker depends on factors such as the strength of the cryptographic algorithms used, the frequency of checks, and the robustness of the overall system architecture. Challenges in implementing integrity checkers include balancing performance with security and addressing evolving threats. Integrity checkers play a vital role in safeguarding data integrity across various domains. Their continued development and integration into systems are essential for ensuring reliable and secure information management in an increasingly complex digital landscape.**

Keywords— **Unauthorized changes, Data integrity, File modification.**

I. INTRODUCTION

File Integrity Monitoring (FIM) is a security practice that involves continuously tracking and detecting changes to files and system configurations to ensure they remain in a desired state. This process helps organizations protect sensitive data, maintain system integrity, and comply with security standards and regulations.

FIM works by creating a baseline of approved file states, including attributes like size, permissions, and cryptographic hash values. The system then monitors these files in real-time or periodically, alerting administrators if unauthorized changes are detected. These changes could be an indicator of malicious activity such as malware infections, unauthorized user actions, or insider threats.

File Integrity Monitoring (FIM) is a crucial security practice of the designed to detect unauthorized changes to files and system configurations. It works by establishing a baseline of approved file states, including key attributes like size, permissions, and hash values, and then monitoring for deviations from this baseline. FIM alerts administrators when any unauthorized or suspicious changes are detected, which could indicate potential security threats, such as malware infections or insider tampering. This tool plays a significant role in helping organizations maintain compliance with regulatory standards like PCI DSS, HIPAA, and SOX, which require rigorous monitoring of file integrity. Additionally, FIM aids in early detection of intrusions, preserves operational stability by identifying unexpected changes, and provides valuable data for forensic analysis in the event of a security breach. By continuously monitoring critical files and system components, FIM enhances both the security and stability of an organization's IT environment.

File Integrity Monitoring (FIM) not only serves as a protective measure against security threats but also contributes to overall system management and compliance. notifies administrators for further investigation. FIM can be deployed in two modes: agent- based and agentless. Agent-based FIM involves installing software agents on the monitored systems, providing detailed, real-time alerts and greater control, whereas agentless FIM scans network systems without installing any agents but may have a slower detection process.

Location-based file integrity monitoring is a security approach that tracks the integrity of files in specific directories or areas within a system, focusing on the most critical or sensitive files. By monitoring changes within defined locations, it can quickly detect unauthorized modifications, additions, or deletions.

This targeted approach optimizes system resources, reducing the overhead of monitoring every file on the system. When combined with preventive measures, such as access controls and automated responses, it can block or alert on suspicious activity before damage occurs. This method helps ensure that only authorized users can access or alter essential files, enhancing overall system integrity and security.

II. LITERATURE SURVEY

A literature survey on File Integrity Monitoring (FIM) involves reviewing various academic, technical, and industry resources to gather knowledge on the development, implementation, and significance of FIM in cybersecurity. This review highlights key concepts, tools, trends, and challenges in the field, providing a solid understanding of the role FIM plays in modern security strategies.

The literature on File Integrity Monitoring underscores its critical role in maintaining system and data integrity, detecting unauthorized changes, and ensuring regulatory compliance. FIM has evolved significantly, from early change detection tools to comprehensive solutions that integrate with broader security frameworks. As organizations increasingly move to the cloud and adopt new technologies, FIM research is focusing on adaptability, reducing false positives, and leveraging AI to improve accuracy. However, challenges remain in scalability and performance, particularly in large, distributed, or virtualized environments. Future research is likely to explore these challenges further, along with the development of next- generation FIM tools capable of addressing evolving security threats.

III. SYSTEM ANALYSIS

A. Existing System

Existing File Integrity Monitoring (FIM) systems are vital tools for detecting unauthorized changes to files and ensuring the integrity of critical data. These systems work by creating a baseline of file states, including attributes like size, permissions, and cryptographic hash values, and then monitoring for any deviations from this baseline. When changes occur, such as file modifications, additions, deletions, or permission changes, FIM systems alert administrators to investigate. Current FIM solutions can be deployed in two ways: agent-based, where software agents are installed on endpoints for real-time monitoring, or agentless, which uses remote scanning for detection. These systems often integrate with broader security tools like Security Information and Event Management (SIEM) platforms to correlate file changes with other security events. Additionally, FIM plays a crucial role in regulatory compliance, helping organizations meet standards like PCI DSS, HIPAA, and SOX by monitoring critical files and generating audit trails. However, despite their effectiveness, existing FIM systems face challenges such as generating false positives, scalability issues in large environments, and difficulties in monitoring cloud-based and encrypted files. Popular tools like Tripwire, OSSEC, and SolarWinds Security Event Manager** represent the current state of FIM technology, balancing file.

B. Proposed System

The proposed File Integrity Monitoring (FIM) system addresses the limitations of traditional solutions by integrating advanced technologies like machine learning, blockchain, architectures to improve accuracy, scalability, and adaptability. Unlike existing systems, the proposed FIM system reduces false positives by using machine learning to distinguish between legitimate changes, such as system updates, and potential security threats. It is built to support cloud environments and virtualized infrastructures, allowing real- time monitoring across hybrid and multi-cloud setups. Additionally, the use of blockchain technology ensures file integrity by storing hash values in an immutable ledger, providing tamper-proof evidence of file states. This system also enhances incident detection through predictive alerts, identifying suspicious patterns before security breaches occur. Its context-aware monitoring ensures that changes are evaluated based on factors like time, user roles, and related activities, which helps prioritize critical alerts. The proposed system is highly scalable and optimized for performance, capable of handling large, distributed networks without causing system slowdowns. It includes a centralized management dashboard for monitoring and compliance reporting, simplifying regulatory adherence to standards like PCI DSS and HIPAA. By supporting encrypted file monitoring and delivering proactive security, this FIM system offers a robust solution for safeguarding critical files in modern, dynamic IT environments.

The proposed File Integrity Monitoring (FIM) system builds upon modern advancements to create a more efficient, intelligent, and adaptable solution for monitoring file changes and ensuring system integrity. One of the key innovations is the integration of machine learning (ML), which enables the system to learn normal file behavior over time and distinguish between legitimate changes (such as scheduled updates or maintenance) and suspicious alterations. This capability drastically reduces the number of false positives—a common issue in traditional FIM systems—allowing security teams to focus on real threats rather than being overwhelmed by benign alerts. The ML algorithms also enhance the system's ability to adapt to evolving environments, providing more accurate and timely alerts.

IV. METHODOLOGY

Location-Based File Integrity Monitoring (FIM) combines geolocation and file integrity monitoring techniques to enhance cybersecurity. It aims to monitor and verify the integrity of files based on the physical or logical location from which files are accessed, modified, or moved. This methodology not only detects unauthorized changes to files but also helps prevent tampering by restricting file interactions to specific locations. Here's a breakdown of a typical methodology:

1. Identify and Classify Critical Files

- Identify sensitive files and folders that require protection.
- Classify files based on their sensitivity and importance (e.g., confidential, high-risk, or mission-critical).
- Establish a baseline for each file's integrity, including metadata (e.g., hash value, last modified time).

2. Define Authorized Locations

- Specify trusted locations from which each file can be accessed or modified. Locations can be defined based on IP addresses, geolocation data (e.g., specific regions or coordinates), or specific network zones.
- Set different access levels for each location (e.g., full access, read-only, restricted).
- Design access policies for each location to ensure that files are not accessible from unauthorized zones.

3. Implement Geolocation-Based Access Control

- Integrate location-based access control mechanisms that check the user's current location before allowing file access or modifications.
- Use tools or scripts that detect the user's location (e.g., through IP, GPS, or network properties) and allow file interactions only if the location matches pre-defined, trusted zones.

4. Apply File Integrity Monitoring (FIM) Mechanisms

- Set up FIM software that continuously monitors and logs changes to the specified files.
- Generate baseline checksums (e.g., using SHA-256) to compare against current file states.
- Schedule regular scans or enable real-time monitoring to detect and log changes to file contents, attributes, or metadata.
- Record details of unauthorized or unexpected changes, including timestamps, IPs, and user identities.

5. Integrate Location-Based Alerts and Responses

- Create an alerting system that triggers notifications when files are accessed or altered from unauthorized locations.
- Customize alert thresholds for specific files or locations (e.g., generate high-priority alerts for high- risk files accessed from unknown zones).
- Implement automated responses, such as reverting unauthorized changes, locking files, or isolating compromised systems.

6. Prevent Unauthorized Access and Changes

- Use access control lists (ACLs) to restrict access to critical files based on user roles, locations, and devices.
- For unauthorized access attempts, enforce automated prevention techniques, such as blocking the IP address, restricting access to a VPN, or quarantining the device.
- Employ data loss prevention (DLP) tools to prevent the copying or moving of sensitive files to unapproved locations.

7. Logging and Auditing

- Keep comprehensive logs of all file interactions, including access times, locations, and user details.
- Regularly audit the logs to verify adherence to location- based policies and identify suspicious behavior patterns.
- Conduct periodic reviews and audits to assess the effectiveness of the location-based FIM approach and to make adjustments as needed.

8. Implement Machine Learning for Detection

- Use machine learning algorithms to analyze historical access patterns and identify anomalies in file access or modification.
- Train models to flag any deviations from typical behavior, considering location, time, and file type.
- Update models regularly to adapt to changing access patterns and minimize false positives.

9. Test and Validate the System

- Perform penetration testing to validate the security of the FIM system and the accuracy of location-based access controls.
- Simulate various scenarios, including file access from unapproved locations, to ensure that the FIM setup correctly identifies and prevents unauthorized access.

10. Continuous Improvement

- Continuously update and improve the FIM system based on new security threats, access patterns, and organizational needs.
- Regularly review and update the list of authorized locations, file classifications, and baseline settings.
- Engage in ongoing monitoring and evaluation to ensure compliance with security policies and evolving regulatory requirements.

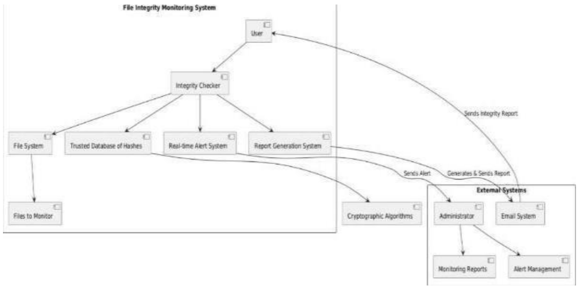
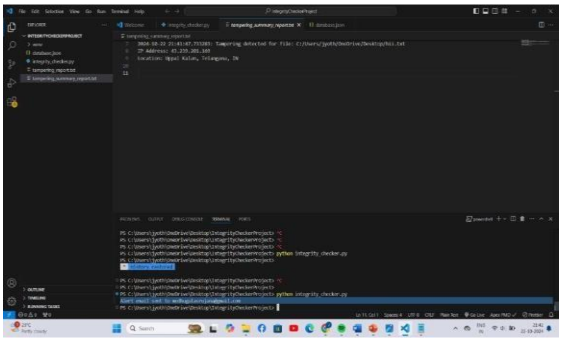


Figure 1: Architecture diagram of FIM flow

This diagram represents the File Integrity Monitoring System architecture, which ensures the integrity of system files using cryptographic algorithms. The Integrity Checker interacts with the file system and a Trusted Database of Hashes to detect discrepancies. It includes a Real-Time Alert System and a Report Generation System to notify users and administrators about unauthorized changes. External systems like the Email System and Monitoring Reports facilitate report dissemination and alert management.

V. RESULTS

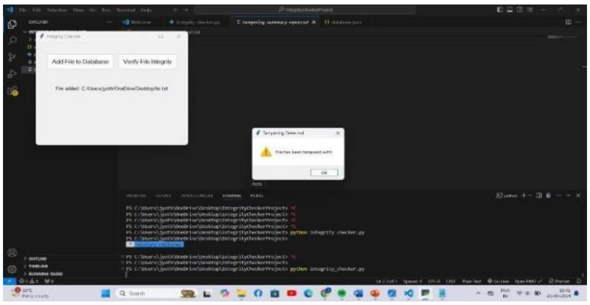


```
2024-10-22 21:41:47.733283: Tampering detected for file: C:/Users/jyoth/OneDrive/Desktop/hii.txt
IP Address: 43.239.201.169
Location: Uppal Kalan, Telangana, IN
```

```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
Alert email sent to [illegible]@gmail.com
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
```

Fig 2: Result Diagram of the execution of FIM

The image describes about the execution and path of the Execution.

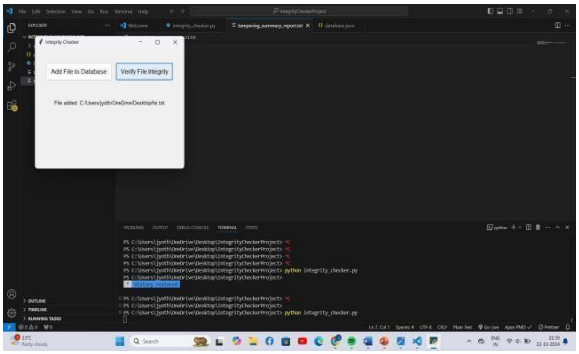


File added: C:/Users/jyoth/OneDrive/Desktop/hi.txt

File has been tampered with!

```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
```

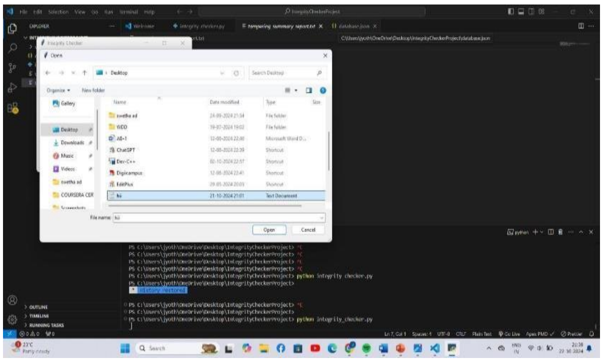
Fig 3: Checking File integrity of file This image defines that file is tampered or not and alert to admin



File added: C:/Users/jyoth/OneDrive/Desktop/hi.txt

```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
```

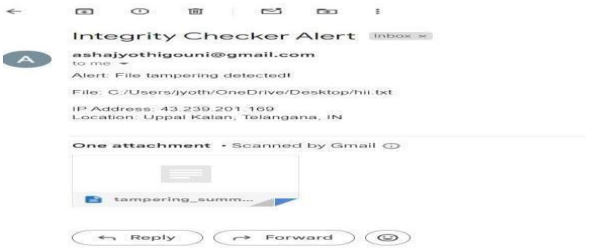
Fig 4: The user verifying the integrity of the file



C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject\database.json

```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity_checker.py
```

Fig 5: To add the file to database we add the file database



Integrity Checker Alert

ashajyothigouni@gmail.com
to me

Alert: File tampering detected!

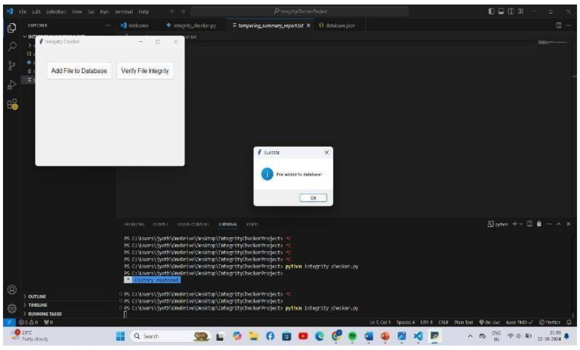
File: C:/Users/jyoth/OneDrive/Desktop/hii.txt

IP Address: 43.239.201.169
Location: Uppal Kalan, Telangana, IN

One attachment • Scanned by Gmail

tampering_summ...

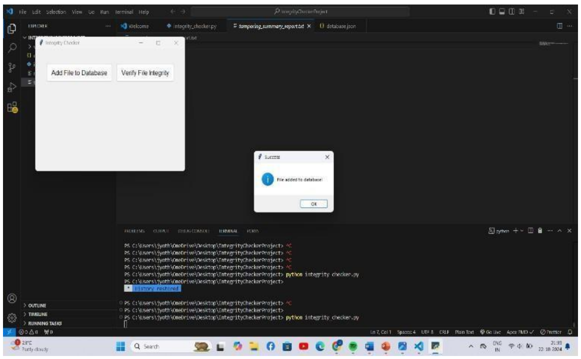
Fig 6: File verified and sent notification to admin mail with report.



File added to database!

```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 *  History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity checker.py
```

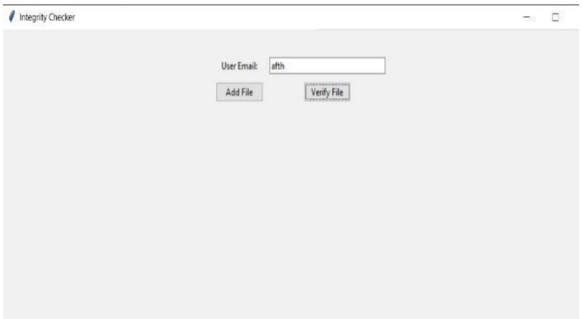
Fig 7: This image shows the pop up of a file that file added to database



File added to database!

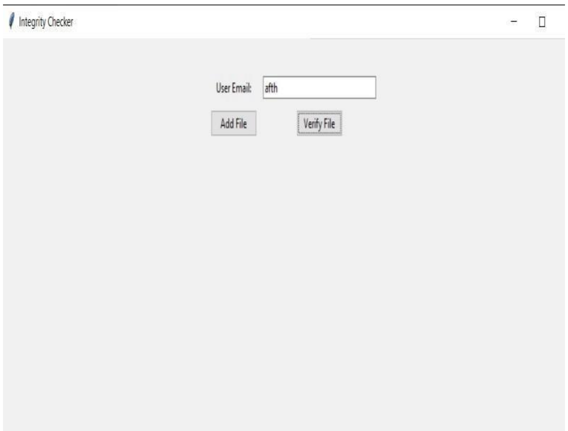
```
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity checker.py
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
 *  History restored

PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> ^C
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject>
PS C:\Users\jyoth\OneDrive\Desktop\IntegrityCheckerProject> python integrity checker.py
```

Fig 8: Adding another file to admin database to monitor the file



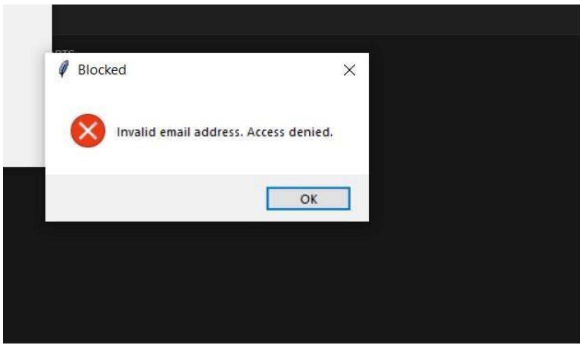
User Email: afth

Fig 9: User entering the email to verify or add the file to database



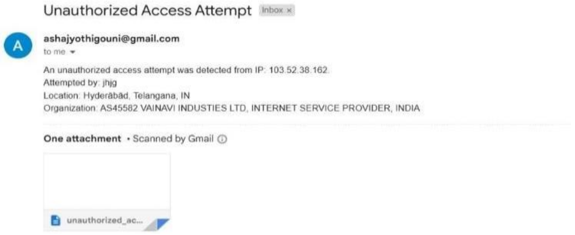
User Email: afth

Fig10: Entering user mail to verify the file in database



Invalid email address. Access denied.

Fig 11:The image displays a pop-up error message from the due to an invalid email address.



Unauthorized Access Attempt

ashajyothigouni@gmail.com

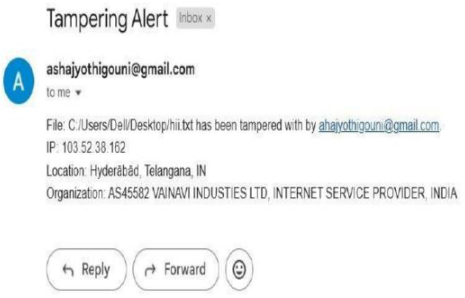
to me

An unauthorized access attempt was detected from IP: 103.52.38.162.
Attempted by: jhjg
Location: Hyderābād, Telangana, IN
Organization: AS45582 VAINAVI INDUSTIES LTD, INTERNET SERVICE PROVIDER, INDIA

One attachment • Scanned by Gmail

unauthorized_ac...

Fig 12:The image shows an email alert about an unauthorized access attempt from IP 103.52.38.162, located in Hyderabad, India,



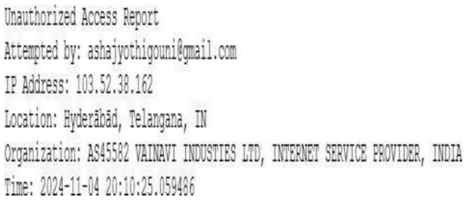
Tampering Alert

ashajyothigouni@gmail.com

to me

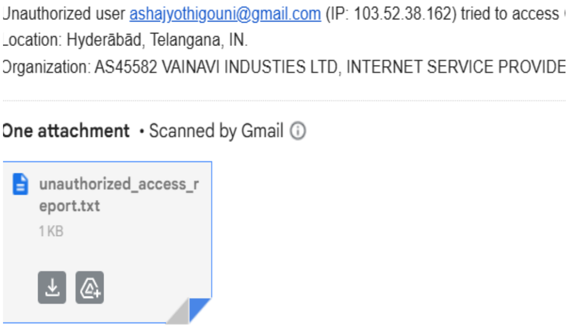
File: C:/Users/Dell/Desktop/hii.txt has been tampered with by ashajyothigouni@gmail.com
IP: 103.52.38.162
Location: Hyderābād, Telangana, IN
Organization: AS45582 VAINAVI INDUSTIES LTD, INTERNET SERVICE PROVIDER, INDIA

Fig 13: It shows the email notification about the ip address of tampering happened and the location.



```
Unauthorized Access Report
Attempted by: ashajyothigouni@gmail.com
IP Address: 103.52.38.162
Location: Hyderābād, Telangana, IN
Organization: AS45582 VAINAVI INDUSTIES LTD, INTERNET SERVICE PROVIDER, INDIA
Time: 2024-11-04 20:10:25.059486
```

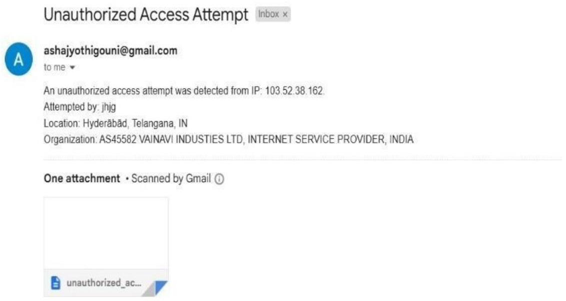
Fig 14: Information of report generated of tampering alert send to admin



Jnauthorized user ashajyothigouni@gmail.com (IP: 103.52.38.162) tried to access
_ocation: Hyderābād, Telangana, IN.
Drganization: AS45582 VAINAVI INDUSTIES LTD, INTERNET SERVICE PROVIDE

One attachment • Scanned by Gmail

unauthorized_access_report.txt
1 KB



Unauthorized Access Attempt

ashajyothigouni@gmail.com

An unauthorized access attempt was detected from IP: 103.52.38.162.
Attempted by: jhjg
Location: Hyderābād, Telangana, IN
Organization: AS45582 VAINAVI INDUSTIES LTD, INTERNET SERVICE PROVIDER, INDIA

Fig 15: Unauthorized access attempt send to admin and location Ip address.

VI. CONCLUSION

In conclusion, Location-Based File Integrity Monitoring (FIM) an innovative extension of traditional file monitoring that adds on critical context by incorporating geographic data.This added layer of the location-based intelligence allows organizations to the better understand a manage who is accessing files and from where, improving their ability to detect, analyze, and respond to unauthorized access or potential security breaches.

VII FUTURE SCOPE

Intelligent Anomaly Detection: AI and machine learning can be used to establish baseline “normal” access behaviors for users and locations, allowing FIM to detect anomalies that go beyond basic location data. For example, if a user accesses a file from a permitted location but exhibits unusual behavior (e.g., accessing files at abnormal times or frequencies), the system can trigger alerts.

Behavior Prediction: Future FIM systems could use predictive analytics to forecast potential unauthorized file changes based on location patterns, enhancing proactive security measures.

Mobile Device Integration: As more employees access data from mobile devices, FIM systems could leverage GPS data from mobile devices to provide a more granular and accurate location-based context for monitoring file integrity.

Smart Contracts for Automated Responses: Smart contracts could be programmed to trigger specific actions (e.g., blocking access or notifying administrators) if file integrity is compromised from an unauthorized location, enhancing automated response mechanisms.

REFERENCES

[1] Kim, G. H., & Spafford, E. H. (1994). "The design and implementation of tripwire: A file system integrity checker." Proceedings of the 2nd ACM Conference on Computer and Communications Security. This paper introduces Tripwire, a widely used FIM tool, detailing its design and principles for integrity checking.

[2] Moustafa, N., & Slay, J. (2015). "The significant features of the UNSW-NB15 and the KDD99 datasets for network intrusion detection systems." This paper introduces Tripwire, a widely used FIM tool.

[3] Moustafa, N., & Slay, J. (2015). "The significant features of the UNSW-NB15 and the KDD99 datasets for network intrusion detection systems." 4th International Workshop on Building Analysis Datasets and Gathering Experience Returnsfor Security(BADGERS),IEEE.

[4] Almiani, M. (2020). "Geolocation-based Security Mechanism for Sensitive Information Protection." International Journal of Information Security Science, 9(3), 154–163.

[5] This study discusses the application of geolocation in Vance, A., & Siponen, M. (2012). "IS security policy violations: A rational choice perspective." Journal of Organizational and End User Computing (JOEUC), 24(1), 21-41.

[6] This research considers contextual data like location as a factor in security violations and policy compliance, relevant for developing adaptive FIM systems that factor in location.

[7] Monitoring sensitive data access and preventing unauthorized. Changes Wagner, C., & Wudi, C. (2018). "Enhancing SIEM with user and entity behavior analytics to detect advanced attacks." Computers & Security, 70, 376-390.

[8] This paper explores the use of user and entity behavior analytics, including location-based context, for advanced threat detection, which can be applied to FIM systems. Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., & Glezer, C. (2010). "Google Android: A comprehensive security assessment." IEEE Security & Privacy, 8(2), 35-44.

[9] This paper covers context-aware security, including location-based access control, in mobile environments. Although it’s mobile-specific, the techniques discussed can inspire similar measures in FIM solutions. Vance, A., & Siponen, M. (2012). "IS security policy violations: A rational choice perspective." Journal of Organizational and End User Computing (JOEUC), 24(1), 21-41.

[10] This research considers contextual data like location as a factor in security violations and policy compliance, relevant for developing adaptive FIM systems that factor in location.