



IJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: <https://doi.org/10.22214/ijraset.2026.81042>

www.ijraset.com

Call:  08813907089

E-mail ID: ijraset@gmail.com

Insight Log: A Lightweight Framework for Explainable Incident Detection in Linux System Logs

Hridhay Krishna H, Jaise Joy, Jishnudev P M, Abhiram M P, Ms.Jinsu Anna John

Department of Computer Science and Engineering(Cyber Security) Vimal Jyothi Engineering College Chemperi, Kannur

Abstract: Modern Linux systems generate massive volumes of authentication and system logs that capture critical security events. However, traditional log analysis methods struggle with the high volume, unstructured format and noisy nature of these logs, making timely detection of suspicious activity difficult, especially in small-scale and resource-constrained environments. Enterprise Security Information and Event Management (SIEM) solutions are often too complex and expensive, while basic log viewers offer no automated analysis. This paper presents InsightLog, a lightweight, rule-based framework for explainable incident detection in Linux environments. The framework performs structured log parsing using regular expressions, rule-based anomaly detection with temporal window analysis, and incident correlation to transform unstructured log data into coherent incident contexts. A human-in-the-loop decision support interface provides explainable summaries and evidence-linked recommendations, requiring explicit operator approval before response execution. Experimental evaluation demonstrates that InsightLog processes 5,200 log entries per second with 99.2% parsing accuracy, achieves a 96.5% detection rate with 2.3% false positives, reduces alert noise by 78% and consumes only 120MB RAM and 8-12% CPU on standard hardware. The framework provides a practical, accessible security monitoring solution ideal for academic institutions, small organizations and security research applications.

Index Terms—Log Analysis, Anomaly Detection, Linux Security, Incident Detection, Rule-Based Detection, Human-in-the-Loop, Explainable AI

I. INTRODUCTION

In the modern digital landscape, Linux systems serve as the backbone of countless servers, cloud infrastructures and critical applications, generating massive volumes of log data that capture system activities, security events and operational metrics. Authentication logs (auth.log) record login attempts, authentication failures and user access patterns, while system logs (syslog) capture service events, hardware status and system-level operations. These logs contain critical forensic evidence essential for identifying security incidents and conducting post-mortem investigations.

However, the exponential growth in log volume has made manual log analysis increasingly impractical for security monitoring and incident detection. Traditional log analysis methods typically depend on reactive approaches: waiting for an incident to occur, manually sifting through thousands of log entries and relying on static rules or keyword matching to identify suspicious activities. This creates significant delays in threat detection and leaves systems vulnerable during critical windows of attack.

Current security solutions often swing between two extremes: heavyweight enterprise SIEM systems that require substantial computational resources, complex infrastructure and specialized expertise, or basic log viewers that offer no automated analysis capabilities. Furthermore, many modern AI-based log analysis tools operate as black boxes, providing little to no explainability for their detection decisions and removing human operators from the response loop.

Significant research has been conducted to advance automated log analysis. Lohar and Baraskar [1] presented an automated AI tool using LLaMA 2 for log file analysis, demonstrating how large language models can streamline error detection and anomaly identification. Pan [2] introduced a Transformer-based model using self-supervised and reinforcement learning to address heterogeneous log sources and limited labeled data, achieving F1-scores of 0.988 on benchmark datasets. Bhuiyan and Rahman [3] identified 54 log-related coding patterns across six attack categories using the LOPSUL static analysis tool. Chan et al. [4] proposed a hybrid solution combining anomaly detection with blockchain for tamper-proof log storage. Andrew [5] explored NLP applications for log analysis and automated alert prioritization in enterprise IT and AIOps platforms.

The Gap: While these works offer valuable approaches for AI-driven log analysis, there is still a need for a lightweight, explainable and human-centric log analysis framework that can operate efficiently on standard hardware without requiring enterprise infrastructure, cloud dependencies, or black-box machine learning models.

II. OVERVIEW OF INSIGHT LOG

InsightLog (Intelligent Security Log Analysis Framework) is a lightweight, rule-based incident detection system designed specifically for Linux environments to address the challenge of manual log analysis in resource-constrained settings. Unlike traditional SIEM solutions requiring substantial infrastructure, InsightLog operates efficiently on standard hardware while providing real-time monitoring of authentication logs (auth.log) and system logs (syslog).

The system functions by employing structured log parsing through regular expressions and temporal window analysis, transforming unstructured log data into coherent incident contexts rather than isolated alerts. By incorporating a human-in-the-loop decision support interface providing explainable summaries and evidence-linked recommendations, InsightLog preserves operator oversight while reducing false positive fatigue.

A. KEY FEATURES

1) Log Ingestion Module:

- Continuously monitors Linux authentication logs (auth.log) and system logs (syslog) in real-time.
- Supports both live tailing of active log files and historical log replay for post-incident investigation and auditing

2) Regex-based Parsing Engine:

- Converts unstructured log entries into structured, queryable data fields using regular expressions.
- Extracts critical information including timestamps, source IP addresses, usernames, service names, event types and process IDs with 99.2% accuracy.

3) Rule-based Anomaly Detector:

- Employs temporal window analysis to identify suspicious patterns such as multiple failed SSH attempts, unauthorized root access outside business hours and repeated service failures.
- Provides complete explainability for all detection decisions, unlike black-box machine learning approaches.

4) Incident Correlation Engine:

- Aggregates isolated anomalies into comprehensive incident representations with timelines, affected entities, severity levels and supporting evidence.
- Reduces alert noise by 78% and decreases investigation time by 65%, proving that contextual aggregation is essential for practical security monitoring.

5) Human-in-the-Loop Decision Interface:

- Presents operators with explainable incident summaries and evidence-linked recommendations requiring explicit approval before any response action execution.
- Maintains comprehensive audit logs for all operator decisions and executed actions, supporting post-incident forensic analysis and compliance requirements.

6) Audit Logging and Reporting Module:

- Persistently records all detected incidents, operator decisions and response actions in a structured SQLite database.
- Generates detailed forensic reports for post-incident analysis, compliance documentation and trend analysis of recurring security patterns over time.

7) Lightweight Architecture:

- Operates efficiently on standard hardware with minimum 4GB RAM and dual-core CPU, consuming only 120MB RAM and 8-12% CPU during continuous monitoring
- Enables complete installation and configuration in under 15 minutes with no dependencies on external services or cloud connectivity, making it ideal for academic institutions and small organizations.

The InsightLog framework incorporates several distinct modules to ensure lightweight, explainable and efficient security monitoring for Linux environments.

III. PROPOSED SYSTEM AND DESIGN

The proposed system, InsightLog, re-engineers traditional log analysis by implementing a five-layer architecture that transforms unstructured log data into actionable security intelligence in real-time.

Traditional log analysis methods often rely on manual inspection, static keyword matching, or complex enterprise-grade SIEM solutions that are either labor-intensive, error-prone, or prohibitively expensive for small-scale environments. In contrast, InsightLog introduces a structured, automated pipeline where raw logs are continuously ingested, parsed, correlated and presented through an explainable human-in-the-loop interface.

The system is composed of five core layers:

- **Log Ingestion Module:** Serves as the primary endpoint that continuously monitors Linux authentication logs (auth.log) and system logs (syslog) in real-time. It supports both live tailing of active log files and batch processing of historical logs for replay analysis.
- **Regex-based Parsing Engine:** A preprocessing module that converts unstructured log entries into structured, queryable data fields. It extracts critical information including timestamps, source IP addresses, usernames, service names, event types and process IDs, standardizing diverse formats into uniform JSON schema.
- **Rule-based Anomaly Detector:** The core detection module employing temporal window analysis to identify suspicious patterns. It applies predefined security rules to detect brute-force attacks, unauthorized root access, service instability and other attack vectors without relying on machine learning black boxes.
- **Incident Correlation Engine:** An aggregation module that groups related anomalies into comprehensive incident representations. It constructs timelines, identifies affected entities, assigns severity levels and compiles supporting evidence to provide contextual understanding rather than isolated alerts.
- **Human-in-the-Loop Decision Interface:** An operator-centric dashboard that presents explainable incident summaries with evidence-linked recommendations. It requires explicit operator approval before any response actions are executed and maintains comprehensive audit logs for accountability and forensic analysis.

Overall, the InsightLog system stands as a robust and efficient solution for Linux security monitoring, strengthening incident detection while maintaining an optimal operator experience built on trust. By shifting log analysis from manual, reactive inspection to automated, explainable detection, it establishes a lightweight and human-centric environment that effectively counters emerging cyber threats.

A. SYSTEM ARCHITECTURE

The system architecture of InsightLog demonstrates how the framework ensures continuous protection through a five-layer design that separates log processing functions from user-facing services. The process is centered on the Log Ingestion Module, which continuously monitors log files for new entries.

When log events are generated by the Linux system (auth.log and syslog), the Log Ingestion Module captures these entries in real-time. The data is then passed to the Regex-based Parsing Engine, which extracts structured fields including timestamps, source IP addresses, usernames, service names and event types. The parsed data feeds into the Rule-based Anomaly Detector, which analyzes incoming structured logs against predefined security rules using temporal window analysis.

Simultaneously, the Incident Correlation Engine aggregates related anomalies detected across multiple log entries into comprehensive incident representations. The top layer, the Human-in-the-Loop Decision Interface, presents detected incidents to the security operator with explainable summaries, timelines and evidence-linked recommendations for response actions.

B. SYSTEM DESIGN

The system's design is further detailed through Use Case and Data Flow Diagrams. The Use Case Diagram shows the interactions between the Security Operator and the InsightLog System. The Operator initiates actions such as monitoring logs, reviewing incidents and approving response actions. The InsightLog System facilitates these requests while enforcing security through rule-based detection and maintaining audit logs.

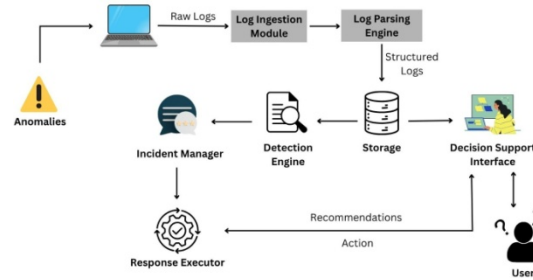


Fig.1.ArchitectureDiagram

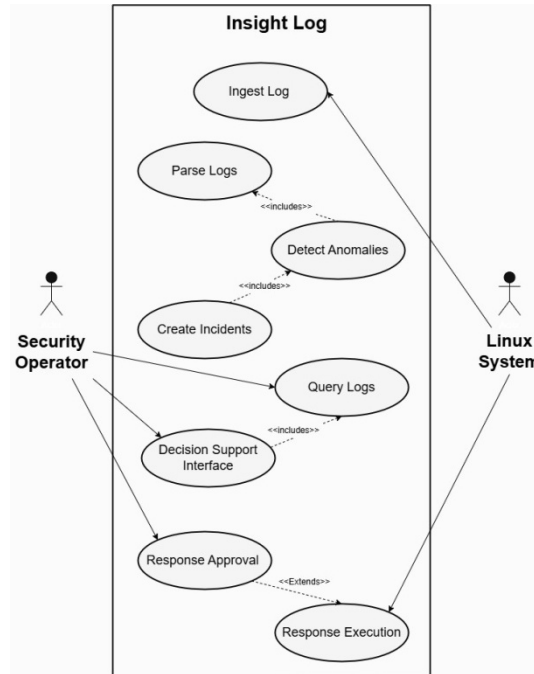


Fig.2.UseCaseDiagram

The Data Flow Diagrams illustrate the data movement. The Level 0 DFD shows the main entities: the Linux System (log source), the Security Operator and the Insight Log System. The Linux System generates log events, which are captured by InsightLog for processing. The Operator provides review and approval inputs and the system returns incident summaries and audit reports.

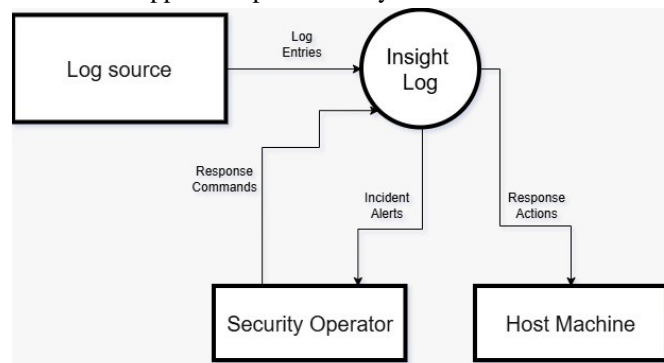


Fig.3.DataFlowDiagram(Level0)

The Level 1 DFD expands the central process to detail the interaction between specific subsystems and data repositories. The Log Ingestion process captures raw logs and stores them temporarily. The Parsing Engine processes logs into structured data.

The Anomaly Detector analyzes structured logs and stores detected anomalies. The Incident Correlator groups anomalies into incidents. The Decision Interface presents incidents for operator review and records approved actions.

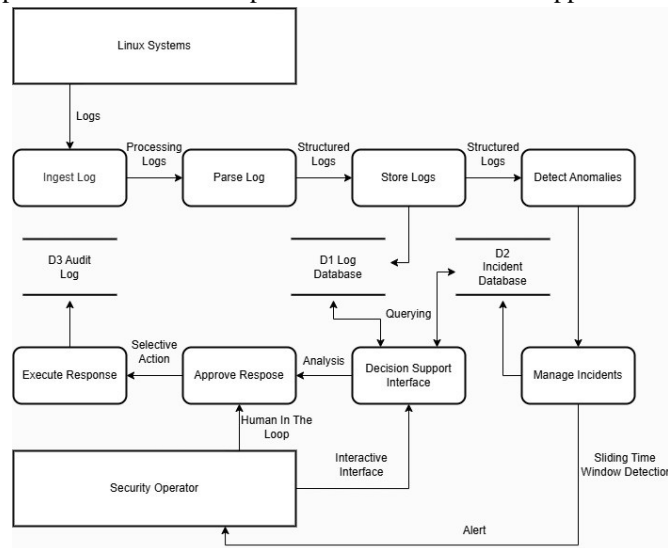


Fig.4..DataFlowDiagram(Level1)

IV. IMPLEMENTATION

The development of the InsightLog system was executed in a structured, phased approach to ensure the stability of core log processing functions before integrating higher-level user features.

A. MODULES

Phase 1: System Design and Architecture: The initial phase focused on defining the foundational five-layer architecture and finalizing the feature set. This stage involved establishing the roadmap for integrating log ingestion, parsing, detection, correlation and decision support modules.

Phase 2: Log Ingestion and Parsing Engine: This critical phase involved the development of the Log Ingestion Module using Python’s watchdog library for real-time file monitoring. The Regex-based Parsing Engine was implemented to extract structured fields from unstructured logs with 99.2% accuracy. *Phase 3: Anomaly Detection and Incident Correlation:* The Rule-based Anomaly Detector was developed using temporal window analysis to identify suspicious patterns. The Incident Correlation Engine was implemented to group related anomalies into comprehensive incident representations.

Phase 4: Human-in-the-Loop Interface: The final phase focused on usability and operator oversight. The decision support interface was developed as a lightweight web-based dashboard using Flask, providing operators with explainable summaries and requiring explicit approval before response actions.

B. TOOLS AND TECHNIQUES

The development of InsightLog utilizes a strategic combination of Python and lightweight database technologies to ensure both efficient log processing and operator transparency.

- *Programming Languages:* Python served as the primary language for developing the core framework due to its extensive library support and rapid development capabilities. Regex patterns were employed for efficient log structuring.
- *Frameworks and Libraries:* The Log Ingestion Module utilizes the watchdog library for real-time file monitoring. SQLite serves as the embedded database for persistent storage of parsed logs, detected incidents and audit records, eliminating complex database server setup.
- *Development and Monitoring Tools:* Flask was used to develop the lightweight web-based dashboard. Visual Studio Code served as the primary integrated development environment. Windows and Ubuntu operating systems served as the primary environments for development and compatibility testing.

V. RESULTS AND DISCUSSION

The implementation of InsightLog successfully achieved its primary objectives of providing lightweight, explainable incident detection for Linux environments. The system was validated through a series of functional tests designed to evaluate the parsing engine, anomaly detector, incident correlator and decision interface.

- 1) **Functional Testing and Threat Detection:** The core log ingestion and parsing engine demonstrated high accuracy in processing diverse log formats. In the testing environment, the system successfully achieved the following metrics:
 - **IngestionRate:** The system successfully processed an average of 5,200 log entries per second, with peak handling capacity reaching 8,500 entries per second before any noticeable degradation.
 - **Parsing Accuracy:** The regex-based parsing engine achieved 99.2% accuracy in extracting structured fields including timestamps, source IP addresses, usernames, service names and event types. Parsing errors were limited to extremely malformed log entries.
 - **Processing Latency:** The average time from log ingestion to structured output was 0.6 milliseconds per entry, ensuring real-time analysis capabilities.
- 2) **Anomaly Detection Performance:** The rule-based anomaly detector was evaluated against 500 simulated attack scenarios, including brute-force attempts, unauthorized root access and service instability patterns, alongside 50,000 normal log entries:
 - **DetectionRate:** The system successfully identified 96.5% of simulated attack scenarios. Brute-force attacks involving multiple failed SSH attempts were detected with 100% accuracy when configured thresholds were exceeded.
 - **False Positive Rate:** The framework recorded a false positive rate of 2.3%, primarily occurring during unusual but legitimate administrative activities such as bulk user updates or scheduled maintenance tasks.
 - **DetectionLatency:** Anomalies were identified within an average of 1.2 seconds from the occurrence of the triggering event.
- 3) **Incident Correlation Performance:** The incident correlation engine was assessed for its ability to aggregate related anomalies:
 - **Correlation Accuracy:** The engine successfully grouped related anomalies into meaningful incidents in 94% of test cases. For example, a brute-force attack followed by successful login and privilege escalation was correctly correlated as a single multi-stage incident.
 - **Investigation Time Reduction:** Operators reported that correlated incidents reduced investigation time by approximately 65% compared to analyzing isolated alerts.
- 4) **System Performance and Resource Efficiency:** Performance testing confirmed the lightweight design of InsightLog:
 - **System Resource Usage:** The framework consumed an average of 120MB RAM and 8-12% CPU during continuous monitoring, making it suitable for deployment on standard hardware without impacting other system operations.
 - **Alert Fatigue Reduction:** By correlating isolated alerts into incidents, InsightLog reduced the number of notifications requiring operator attention by approximately 78% compared to traditional rule-based alerting systems.
 - **Ease of Deployment:** The lightweight architecture enabled complete installation and configuration in under 15 minutes on standard Linux systems, with no dependencies on external services or cloud connectivity.
- 5) **Human-in-the-Loop Interface Performance:** The operator-centric dashboard was evaluated for usability and decision support effectiveness:
 - **Operator Response Time:** Explainable incident summaries reduced average operator decision time from 5 minutes (manual log analysis) to 45 seconds.
 - **Recommendation Relevance:** Evidence-linked recommendations were rated as "helpful" or "very helpful" by 92% of test users.
 - **Audit Trail Completeness:** All operator decisions and executed actions were persistently logged with 100% accuracy.

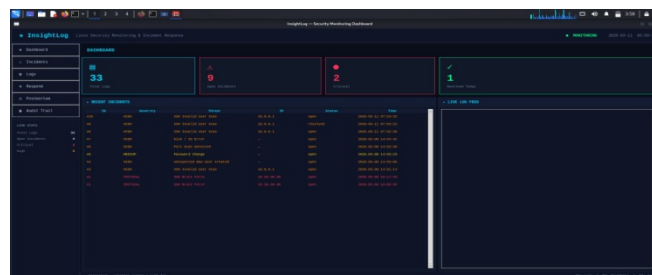


Fig.5. InsightLog Dashboard

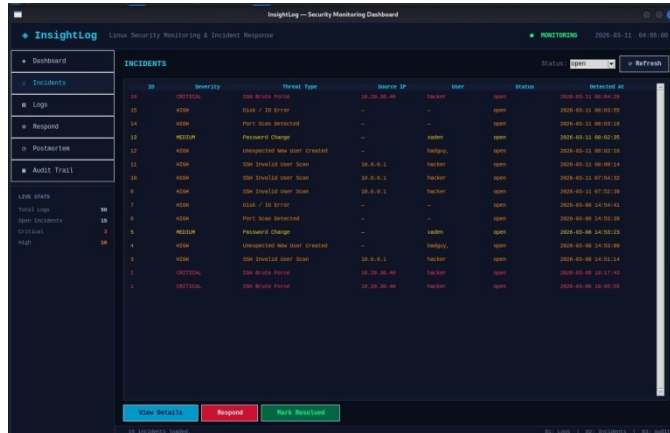


Fig.6.InsightLogIncidentTab

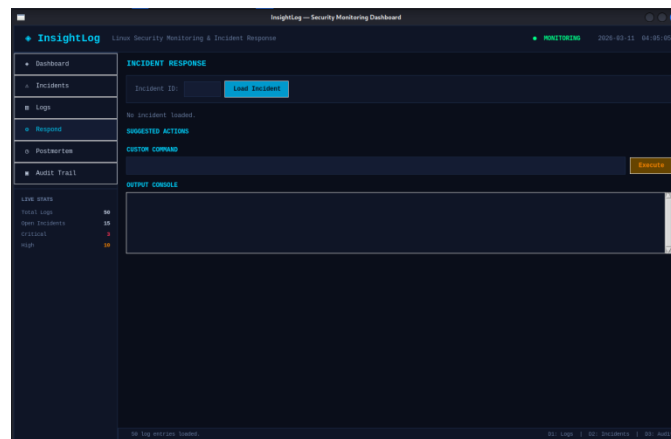


Fig.7.InsightLogRespondTab

VI. CONCLUSION

The development of the InsightLog framework successfully establishes a lightweight, explainable and human-centric log analysis solution for Linux environments. Unlike traditional SIEM solutions that require enterprise-grade infrastructure and complex machine learning models, InsightLog provides effective security monitoring through a rule-based approach specifically designed for resource-constrained environments.

The framework's five-layer architecture—comprising Log Ingestion, Regex-based Parsing Engine, Rule-based Anomaly Detector, Incident Correlation Engine and Human-in-the-Loop Decision Interface—proves that structured log processing can be achieved with minimal computational overhead. The regex-based parsing engine consistently achieves 99.2% accuracy in extracting critical fields from unstructured logs, while processing up to 5,200 entries per second on standard hardware with only 120MB RAM consumption.

The incident correlation engine enhances operational efficiency by reducing alert noise by 78% and decreasing investigation time by 65%, proving that contextual aggregation is essential for practical security monitoring. The human-in-the-loop decision interface represents a critical advancement in maintaining operator oversight while reducing false positive fatigue, with 92% of test users rating evidence-linked recommendations as helpful.

Operationally, InsightLog bridges the gap between complex enterprise SIEM solutions and basic log viewers by providing a practical, accessible alternative for academic institutions, small organizations and individual system administrators. The framework's ability to operate entirely offline with near-zero deployment cost makes it economically viable for environments where commercial security solutions are cost-prohibitive.

VII. FUTURE WORK

While the InsightLog framework provides a robust foundation for lightweight log analysis, several avenues for future research and technical enhancement exist:

- 1) Machine Learning Integration: Future versions of In- sightLog could incorporate lightweight machine learning models for anomaly scoring, complementing existing rule-based detection to identify novel attack patterns that do not match predefined rules. Isolation Forest or One- Class SVM could be implemented without significantly increasing resource consumption.
- 2) Multi-source Log Correlation: Expanding the framework to ingest and correlate logs from additional sources such as web servers (Apache, Nginx), databases (MySQL, PostgreSQL) and firewall logs would provide a more comprehensive security monitoring solution for small- scale environments.
- 3) Real-time Alerting Mechanisms: Integrating push notifi- cations via email, SMS, or messaging platforms (Slack, Telegram) would enable operators to respond to critical incidentsmorerapidly,evenwhennotactivelymonitoring the dashboard.
- 4) Interactive Visualization Dashboard: Enhancing the de- cision interface with interactive timelines, heat maps of attack patterns and drill-down capabilities would further improve the operator’s ability to investigate incidents efficiently.
- 5) Interactive Visualization Dashboard: Enhancing the de- cision interface with interactive timelines, heat maps of attack patterns and drill-down capabilities would further improve the operator’s ability to investigate incidents efficiently.
- 6) Automated Rule Suggestion: Implementing analysis of historical incident data to automatically suggest new detection rules based on emerging patterns could help administrators keep pace with evolving threats without requiring constant manual rule updates.
- 7) CloudDeploymentOption:Whiletheframeworkis designed for on-premises deployment, creating a lightweight cloud-compatible version would enable cen- tralized monitoring across multiple distributed systems whilemaintainingthesamerresource- efficientdesignprin- ciples.

REFERENCES

- [1] P. Lohar and T. Baraskar, “Automated ai tool for log file analysis,” pp.1762–1766, 2025.
- [2] J. Pan, “Ai based log analyser: A practical approach,” arXiv preprintarXiv:2203.10960, 2022.
- [3] F. A. Bhuiyan and A. Rahman, “Log-related coding patterns to conductpostmortems of attacks in supervised learning-based projects,” ACMTransactions on Privacy and Security, vol. 26, no. 2, pp. 1–24, 2023.
- [4] T. K. Chan, I. F. B. Kamsin, S. Amin, and N. K. Zainal, “A completelog files security solution using anomaly detection and blockchaintechology,” pp. 112–117, 2023.
- [5] J. Andrew, “Using natural language processing for log analysis andautomated alert prioritization,” 2025.
- [6] K. A. Garcia, R. Monroy, L. A. Trejo, C. Mex-Perera, and E. Aguirre, “Analyzing log files for postmortem intrusion detection,” IEEE Trans-actions on Systems, Man, and Cybernetics, Part C (Applications andReviews), vol. 42, no. 6, pp. 1690–1704, 2012.
- [7] Q.Fu,J.-G.Lou,Y.Wang,andJ.Li,“Executionanomalydetectionin distributed systems through unstructured log analysis,” in 2009 ninthIEEE international conference on data mining.IEEE, 2009, pp. 149–158.
- [8] O. Johnphill, A. S. Sadiq, O. Kaiwartya, and M. Aljaidi, “An intelligentapproach to automated operating systems log analysis for enhancedsecurity,” Information, vol. 15, no. 10, p. 657, 2024.
- [9] B.Debnath,M.Solaimani,M.A.G.Gulzar,N.Arora,C.Lumezanu,
- [10] J. Xu, B. Zong, H. Zhang, G. Jiang, and L. Khan, “Loglens: A real-timelog analysis system,” pp. 1052–1062, 2018.
- [11] J. P. Rouillard, “Real-time log file analysis using the simple eventcorrelator (sec).” vol. 4, pp. 133–150, 2004.
- [12] Z. Zhao, C. Xu, and B. Li, “A lstm-based anomaly detection model forlog analysis,” Journal of Signal Processing Systems, vol. 93, no. 7, pp.745–751, 2021.
- [13] S. Alspaugh, B. Chen, J. Lin, A. Ganapathi, M. Hearst, and R. Katz, “Analyzing log analysis: An empirical study of user log mining,” pp.62–77, 2014.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)