REAL TIME LOG MONITORING SYSTEM WITH ALERTS

Hemalatha M[1], S Bhoomika[2], Shraddha A Vaidya[3], Sindhushree NH[4]
Department of CSE, Don Bosco Institute Of Technology
hemalatha.rmgm@gmail.com, bhoomigowda572@gmail.com,
shraddha23vaidya@gmail.com, sindhuharish311@gmail.com

Abstract

Abstract—Ensuring the reliability of modern software systems demands continuous monitoring and rapid incident response. This Real-Time Log Monitoring System helps you keep a close eye on your applications and infrastructure by collecting, processing, and visualizing log data instantly. It's designed to make debugging faster and improve the overall resilience of your systems. Instead of shifting through logs manually, the system uses smart scanning to detect important events as they happen—and it automatically sends out alerts, so you're always in the loop.Built on the reliable ELK stack (Filebeat, Logstash, Elasticsearch, and Kibana) and deployed with Docker, it's both scalable and easy to run across different platforms. You get centralized logging, real-time dashboards, and alert notifications via email or webhooks.Perfect for developers and system admins, this tool not only boosts operational efficiency but also lays the groundwork for advanced features like AI-powered anomaly detection and integrations with tools like Slack or Telegram.

Keywords

Real-time monitoring, ELK stack, Docker, system resilience, automated alerts, centralized logging, DevOps, log visualization, incident response, anomaly detection.

Introduction

In today's fast-moving world of digital technology, software applications and infrastructure constantly generate huge amounts of log data as they run. These logs hold valuable insights into how systems behave, how users interact with them, when errors occur, and how well things are performing overall.For software engineers, DevOps teams, and system administrators, this log data is a go-to resource. It helps them spot failures, catch unusual activity, and keep an eye on the overall health of their applications.

Traditionally, log monitoring and analysis have been done manually or with limited automation. Engineers often have to dig through stored log files to figure out what went wrong—*after* a problem has already happened.This reactive approach doesn't work well in modern environments like large-scale distributed systems, microservices, or cloud-native setups, where logs are constantly being generated across many different nodes and platforms.When issues aren't caught quickly, it can lead to longer downtimes, slower system performance, and higher operational costs—all of which can seriously impact business and user experience.

To address these challenges, this project proposes the development of a Real-Time Log Monitoring System with Alerts, aimed at delivering proactive and automated log analysis capabilities.The system operates by constantly collecting and processing log information and system monitors log data to detect important events and identify error patterns in real time, and notify relevant stakeholders through immediate alerts using email or webhook integrations. This real-time alerting mechanism significantly reduces mean time to detection (MTTD) and mean time to resolution (MTTR), contributing to more reliable and resilient system behavior.

A strong DevOps toolchain forms the foundation of the system architecture through the integration of the ELK stack.DevOps toolchain, notably the ELK stack— Elasticsearch for efficient storage and search, Logstash for real-time log processing and filtering, and Kibana for intuitive data visualization. To ensure

portability, scalability, and simplified deployment, the entire system is containerized using The use of Docker enables the system to operate effectively across multiple deployment environments including on- premises, cloud-based, and hybrid infrastructures.

Beyond immediate log monitoring and alerting, this project supports the broader goals of observability and continuous improvement. It empowers DevOps practitioners with actionable insights through real time dashboards and centralized log access. Moreover, the system is designed with extensibility The system's extensible design provides a strong basis for integrating advanced features such as AI/ML driven anomaly detection, predictive analytics, and integration with communication platforms like Slack and Telegram for collaborative incident response.

Ultimately, this project represents a practical implementation of modern DevOps practices, aiming to enhance system visibility, accelerate incident response, and maintain high availability in today's complex software ecosystems.

Literature Review

Real-time log monitoring systems enable smooth operations by identifying problems immediately using live tracking of logs. These systems eliminate manual checking requirements while providing integrated dashboards that deliver instant alerts. Contemporary platforms integrate monitoring tools with notification systems to ensure teams receive real-time updates. These systems continue to struggle with managing massive data quantities alongside alert overloads and compatibility issues between systems.

Dinesh S et. al. (2025) introduced a secure chat-based log monitoring system designed for Kubernetes environments. Their system combines ELK stack and Fluentd to enable real-time log centralization and monitoring while providing enhanced security and immediate visibility. This method demonstrates the necessity for immediate analysis capabilities in container-based systems. The need for intricate setup procedures coupled with Kubernetes integration increases deployment difficulties.[1]

Priya et al. (2025) proposed an AI-driven cyber threat detection system integrating log analysis, machine learning, and SIEM tools. Using the CICIDS2017 dataset, they demonstrated effective detection of cyber threats with high accuracy through real-time monitoring and automated alerts.[2]

Zhang and Wang (2025) developed a real-time access data monitoring and early warning system for fusion research using the Elastic Stack. Their system employs multidimensional logging and anomaly detection to enhance MDSplus audit capabilities, improving data security and governance in experimental environments.[3]

Bobde H. et al. (2024) created a smart alert system designed for monitoring server logs, utilizing machine learning and log parsing techniques. Their innovative model automatically identifies anomalies and sends out alerts, ensuring that system issues are recognized promptly. This solution underscores the importance of automation in keeping systems healthy. However, it might face some hurdles, such as alert fatigue and challenges with scalability in data-heavy environments.[4]

Ho E. et al. (2024) put forward some interesting improvements to Write-Ahead Logging (WAL) that focus on making system maintenance easier by optimizing how indexes are handled and segmented. Their prototype designs really boost data management and make logging more efficient. While this approach does come with some performance perks, it might also add a layer of complexity and require a lot of resources during the implementation phase.[5]

M. Mahmoud et al. (2024) introduced a sandbox-independent malware analysis framework using Sysmon and the ELK Stack. It analyzes behavior from 2,800 malware samples in VMs, collecting host and network logs. The framework avoids sandbox fingerprinting and supports detailed threat investigation. The resulting AAU_MalData dataset aids future anti-malware research.[6]

M.S. Ali and Puri D. (2024) took a deep dive into how artificial intelligence can be woven into DevOps workflows. They focused on strategies for automation and predictive modeling to make software delivery smoother and more efficient. While their approach boosts productivity and encourages smart automation, it also raises some ethical questions and depends a lot on having access to top-notch training data.[7]

Shaji and George (2024) provided a comprehensive overview of the Elastic Stack, detailing the roles of Elasticsearch, Logstash, Kibana, and Beats in real-time data analysis. Their work emphasizes the stack's utility in diverse domains including IT monitoring and public data visualization.[8]

Xu S. et al. (2024) created an innovative anomaly detection platform that combines the ELK stack with Kafka. This setup allows for sophisticated processing of log data and the identification of unusual behaviors. The architecture is built to manage large volumes of logs with impressive accuracy. While it shows excellent performance in real-time analytics, it does come with its share of complexity and significant computational requirements.[9]

Lertwuthikarn et al. (2022) proposed a resource-optimized log processing pipeline using Filebeat and Logstash at CERN's ALICE O2 facility. They used factorial experiment designs and regression modeling to tune system parameters, reducing CPU and memory usage while maintaining high log throughput.[10]

Zou Z. et al. (2022) developed a monitoring system that leverages an optimized Isolation Forest algorithm to spot anomalies in Docker containers. This innovative approach allows for automated monitoring and provides a thorough root cause analysis of how containers behave. However, it's worth noting that the system might encounter some computational challenges when deployed on a large scale.[11]

Chen et al. (2020) introduced a Docker-based container log collection system combining ELK Stack and Kafka. Their architecture enables efficient real-time log collection, filtering, and analysis, improving stability and scalability in containerized environments.[12]

Chaudhari et al. (2019) proposed a real-time ELK-based system to monitor critical security incidents. It collects and analyzes logs from 2500 nodes using Filebeat, Logstash, and Kibana. The system detects threats early and supports centralized visualization. This enhances organizational IT security.[13]

Debnath B. et al. (2018) developed LogLens, a real-time log analysis system built on Spark and unsupervised learning techniques. The system adapts to diverse log formats with minimal user input. While it offers flexibility, it requires periodic retraining and may encounter performance limitations due to the underlying Spark architecture.[14]

Gunasekaran R. et al. (2010) designed a web-based framework for real-time system log monitoring using MySQL, allowing administrators to parse logs and access summaries with low overhead. The system performs well in real-time scenarios but provides limited support for detailed trace-level diagnostics.[15]

Reference	Methodology	Algorithm	Key Features	Limitations
[1]	Secure chat application for real-time log monitoring in Kubernetes using ELK + Fluentd	Not focused on AI/ML, WebSockets.	Real-time communication for log monitoring, scalable Kubernetes deployment.	Does not focus on deep analytics or AI-based anomaly detection
[2]	Log-based cyber threat detection using ML, SIEM, rule-based + statistical analysis	Supervised ML, rule-based methods	Real-time anomaly detection, automated alerts, dataset	Large-scale deployment challenges
[3]	Multidimensional audit logging for MDSplus via Elastic Stack	Logstash filters, Elasticsearch indexing	Real-time and offline audit trail monitoring, dual dashboards	Focused only on MDSplus; local audit trails not supported
[4]	ML-based log parsing and real-time alert system using structured/unstructured logs	Rule-based parsing, supervised & unsupervised ML	Customizable alerting , visualization integration	Limited evaluation detail, requires integration with log platforms
[5]	WAL maintenance system with enhancements for resilience and performance	Log segmentation, compaction	Efficient log recovery, reduced resource consumption	High I/O impact from WAL
[6]	Dynamic malware analysis using virtual environments and ELK + Sysmon integration	Log correlation and pattern analysis	Detailed malware behavior logging	May not bypass all evasion techniques, lacks sandbox-based automation
[7]	Integration of AI techniques in DevOps for optimization	NLP, reinforcement learning	Enhanced CI/CD pipelines, intelligent automation	Data quality issues, integration complexity
[8]	Integrated use of Elasticsearch, Logstash, Kibana, and Beats for log handling and visualization	ELK Stack components	Real-time log processing, intuitive dashboards	No AI/ML-based detection, limited to visualization and aggregation
[9]	Integration of ELK Stack with Kafka and deep learning (NF- GAN) for noise filtering and anomaly detection	NF-GAN , deep learning	Real-time anomaly detection, scalable with Kubernetes	High computational requirements, deep learning complexity

[10]	Factorial design to optimize ELK resource use in large-scale log systems	Factorial design, regression modeling	Optimized Filebeat & Logstash for CPU/memory efficiency	Complex configuration, not generalized
[11]	Multidimensional resource monitoring with adaptive interval control using enhanced Isolation Forest	Optimized Isolation Forest with weighted feature selection	Detects container anomalies, log cause analysis	Focused on resource metrics, less effective
[12]	Dockerized log collection system with Filebeat, Kafka, ELK Stack	Logstash , Kafka for queuing	Real-time container log collection, scalable and efficient analysis	No AI-based threat detection
[13]	ELK-based real-time analysis for critical security incidents in large-scale infrastructure	ELK stack with correlation logic	Critical incident detection for CII, centralized logging	Focused on specific incident types, lacks advanced ML/AI
[14]	Real-time log analysis using unsupervised ML on system logs	Stateless and stateful ML algorithms	Automatic anomaly detection, real-time analysis	Spark's broadcasting limitations, memory overhead
[15]	Web-based log analysis framework using MySQL, Apache, and custom parsers for Jaguar XT5	Statistical profiling, regex-based parsing	Real-time log viewing, anomaly indicators	No AI/ML, limited to structured logs

Methodology

The Real-Time Log Monitoring System with Alerts is engineered to provide immediate visibility into system and application behaviors by continuously collecting, processing, and analyzing log data. The methodology integrates various components into a unified architecture that enables real-time insights, automated alerting, and seamless visualization. The system is designed for scalability, security, and operational efficiency.

The system starts its operation by gathering logs from various sources including application, infrastructure, and database logs through the deployment of lightweight agents on containers and servers. The agents track logs locally and transmit data immediately to a central pipeline. The parsing module standardizes log formats and retrieves metadata such as timestamps and log levels before assigning classification tags to create an organized data stream.

The core processing unit receives parsed logs to maintain structured views which enable efficient access through materialized views and system-level file system metrics monitoring. Logs undergo categorization into different types to enable specialized analysis.

The disk-based backend system stores structured entries and metrics which ensures both durability and scalability while facilitating swift indexing together with high-volume data retention and multiple concurrent queries essential for both real-time and historical analytics.

The system continuously does real-time analysis on log streams, generating alerts based on rules or anomaly detection. Alerts for issues like crashes or resource overloads are automatically sent via email or webhooks. To avoid overwhelming you with too many messages, the system groups related alerts and limits how often they're sent—so you get the important info without all the noise.

The system features a user-friendly web interface with interactive dashboards that display logs, real-time metrics, and alerts all in one place. It also includes an application profiling module that helps you monitor performance in real time and get to the root of issues quickly. With powerful filtering and search tools, it's easy to zero in on specific problems and resolve them faster.

The platform is designed to be flexible and easy to extend. It supports integration with third-party tools like RAVEN for enhanced visualization and more detailed operational reporting. All components are containerized, which means deployments are smooth, scalable, and upgrades can happen with minimal downtime.

Security is built in from the ground up, with encrypted data transport and role-based access control to ensure that only the right people have access. The system is also designed to be resilient—it can automatically recover from service failures thanks to its self-healing capabilities. Persistent storage keeps your data safe and intact, even if something goes wrong. Plus, built-in monitoring of both infrastructure and application health allows for proactive maintenance and ongoing performance optimization.

Expected Results

- The log monitoring system automatically identifies critical events by gathering and analyzing logs in real-time.
- It notifies teams email when custom rules cause alerts to be triggered. Fast issue tracking is made possible by a searchable log history and a live dashboard.
- Fast response, less downtime, and increased system reliability are guaranteed by this configuration.

Conclusion

Error detection, incident response, and system observability are all effectively addressed by the Real- Time Log Monitoring System with Alerts. Continuous log tracking, structured analysis, and immediate alerts are made possible by it, which speeds up problem identification and recovery. Its modular design, which includes dashboards, alert systems, and parsing engines, guarantees scalability and user-friendliness. It is perfect for DevOps and IT infrastructure management because of its open-source, extensible design, which allows for future improvements like anomaly detection based on machine learning and integration with collaboration tools.

References

[1] D. S. Dinesh et al., "Deploy a secure chat application for log monitoring in Kubernetes," in Proc. Int. Conf. IDCIoT, 2025, pp. 516–522. doi: 10.1109/IDCIOT64235.2025.10914743.

[2] V. Priya et al., "Cyber threat detection using supervised learning and log analysis," *Proc. ICSSGT*, 2025, pp. 231–236. doi: 10.1109/ICSSGT56824.2025.10232490.

[3] Y. Zhang and X. Wang, "Real-time access data monitoring and early warning system based on ELK Stack for EAST," *Fusion Eng. Des.*, vol. 207, 2023, Art. no. 113587. doi: 10.1016/j.fusengdes.2023.113587.

[4] H. Bobde et al., "Log alert system server log recognition and alert system," Int. J. Trend Sci. Res. Dev., vol. 8, no. 6, pp. 69–78, Dec. 2024.

[5] E. Ho et al., "Tactics for enhancing write ahead log for long-term system maintenance," in Proc. PlatCon, 2024, pp. 56–63. doi: 10.1109/PLATCON63925.2024.10830742.

[6] M. Mahmoud et al., "Enhancing malware analysis through Sysmon and ELK integration," *SSRN*, 2024.

[7] M. S. Ali and D. Puri, "Optimizing DevOps methodologies with AI," in Proc. INOCON, 2024. doi: 10.1109/INOCON60754.2024.10511490.

[8] S. Shaji and B. George, "ELK Stack: Real-time Data Analysis Framework," *Int. J. Comput. Appl. Technol. Res.*, vol. 13, no. 2, pp. 41–46, 2024. doi: 10.7753/IJCATR1302.1004.

[9] Xu.S et al., "Log-based anomaly detection platform based on ELK+Kafka," in Proc. ICCC, 2024, pp. 1928–1935. doi: 10.1109/ICCC62609.2024.10941763.

[10] T. Lertwuthikarn et al., "Performance optimization of the Elastic Stack for log collection at the ALICE O^2 facility," *J. Phys. Conf. Ser.*, vol. 2438, 2023, Art. no. 012060. doi: 10.1088/1742- 6596/2438/1/012060.

[11] Z. Zou et al., "A Docker container anomaly monitoring system based on optimized isolation forest," IEEE Trans. Cloud Comput., vol. 10, no. 1, pp. 134–147, Jan.–Mar. 2022. doi: 10.1109/TCC.2019.2935724.

[12] Chen et al., "Design of docker container log analysis system using ELK and Kafka," *Proc. ICCEA*, 2020, pp. 361–366. doi: 10.1109/ICCEA50009.2020.00078.

[13] S. Chaudhari et al., "Real-time log and traffic monitoring with ELK," *Proc. NGCT*, 2019, pp. 1–10.

[14] B. Debnath et al., "LogLens: A real-time log analysis system," in Proc. IEEE ICDCS, 2018, pp. 1052–1062. doi: 10.1109/ICDCS.2018.00105. [8] R. Gunasekaran et al., "Real-time system log monitoring/analytics framework," Oak Ridge Natl. Lab., 2010. vol. 8, pp. 120042–120051, 2020.

[15] R. Gunasekaran et al., "Real-time system log monitoring/analytics framework," Oak Ridge Natl. Lab., 2010.

Copyright & License:
© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.