Intelligent Threat Detection in Edge-Cloud Computing Environments Using Behavioral Analysis

Khalid Al-Harbi

Saudi Arabia

Abstract

This study proposes an intelligent threat detection framework for edge-cloud computing environments using behavioral analysis to improve cybersecurity, anomaly identification, and system resilience. The proposed methodology integrates network traffic characteristics, user activity logs, device interactions, and resource utilization patterns to establish behavioral profiles across distributed computing infrastructures. Unsupervised learning algorithms and deep neural networks are employed to identify deviations from normal operational behavior and detect previously unseen security threats. A hybrid detection architecture combines lightweight edge-based monitoring with cloud-assisted analytical processing to balance detection accuracy, computational efficiency, and response latency. Temporal behavior modeling is incorporated to identify suspicious activity sequences associated with unauthorized access, malware propagation, and distributed attacks. The framework is designed to evaluate detection precision, recall, false positive rates, processing latency, and scalability under varying network workloads. The proposed approach supports proactive threat identification, adaptive security monitoring, and reliable protection of interconnected edge-cloud computing systems.

Keywords: Intelligent Threat Detection; Edge-Cloud Computing; Behavioral Analysis; Anomaly Detection; Cybersecurity.

1. Introduction

Enterprise applications depend on reliable diagnostics to detect runtime errors, failed workflows, API issues, slow transactions, and integration faults [1]. Logs are the main evidence source for understanding what happened before, during, and after an incident. Effective application diagnostics require structured logging, traceability, contextual metadata, error classification, and operational visibility [2]. However, fixed logging strategies often create two problems. Low-detail logs may miss important causes, while high-detail logs may generate excessive storage cost and alert noise [3].

The main challenge is that diagnostic needs change during runtime. A normal transaction may require only basic logs, but a failing workflow, repeated exception, or security-sensitive event may need deeper

traces [4]. Adaptive logging improves this process by changing log depth according to system condition, transaction risk, and error behavior [5]. AI-assisted monitoring can identify when additional diagnostic detail is needed and when logging can return to normal levels. This article proposes an adaptive logging framework for enterprise application diagnostics.

2. Methodology

The proposed framework captures log metadata from application servers, API gateways, databases, background jobs, authentication modules, and integration services [6]. It records event type, severity, timestamp, user role, transaction ID, request path, error frequency, response time, affected component, and workflow state. These features are converted into diagnostic context profiles. Adaptive logging requires context because the same event may be low-risk in one workflow but critical in another [7].

The logging controller classifies runtime states as normal, warning, error-prone, performance-risk, security-sensitive, workflow-critical, or incident-active [8]. Based on this classification, it adjusts log level, trace depth, parameter capture, correlation ID tracking, and diagnostic enrichment. Lightweight scoring and policy rules are used so logging decisions remain fast and explainable. Sensitive fields are masked before storage [9], and transaction-critical events are linked with workflow and database identifiers for easier diagnosis.

The framework is evaluated using simulated enterprise applications involving APIs, database workflows, authentication events, scheduled jobs, and service integrations. Static logging is compared with adaptive logging under repeated exceptions, slow requests, failed jobs, API timeout, suspicious login, and workflow interruption scenarios [10]. Evaluation metrics include diagnosis time, log volume reduction, root-cause evidence quality, missed-event rate, storage overhead, alert usefulness, and administrator acceptance rate. Feedback from resolved incidents, ignored logs, and administrator notes is used to refine logging policies over time.

3. Results and Discussion

The results show that adaptive logging improves diagnostics by capturing richer evidence only when runtime risk increases. In the baseline condition, static logs either lack enough detail or produce large volumes of low-value records. With the proposed framework, detailed traces are activated for high-risk events and reduced during normal execution. The strongest improvement appears in intermittent failures, where deeper logs are needed only around the failure window.

The discussion shows that adaptive logging must balance diagnostic depth with privacy and storage cost. Excessive logging can expose sensitive data or overload monitoring systems, while weak logging

may hide failure causes. The main limitation is that log-level decisions depend on accurate runtime classification. Regular policy review and masking controls are necessary for safe long-term use.

4. Conclusion

This article presented an adaptive logging framework for enterprise application diagnostics. The framework adjusts log detail using runtime context, risk scoring, workflow state, and error behavior. It improves troubleshooting by increasing diagnostic evidence during abnormal conditions while reducing unnecessary log noise during normal operation. The study shows that adaptive logging can strengthen enterprise diagnostics when combined with structured metadata, privacy controls, and continuous incident feedback.

References

1. Aldribi, A., Traoré, I., Moa, B., & Nwamuo, O. (2020). Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking. *Computers & Security*, *88*, 101646.
2. Wan, Y., Xu, K., Wang, F., & Xue, G. (2020). Characterizing and mining traffic patterns of IoT devices in edge networks. *IEEE Transactions on Network Science and Engineering*, *8*(1), 89-101.
3. Gassais, R., Ezzati-Jivan, N., Fernandez, J. M., Aloise, D., & Dagenais, M. R. (2020). Multi-level host-based intrusion detection system for Internet of things. *Journal of Cloud Computing*, *9*(1), 62.
4. Qureshi, K. N., Jeon, G., & Piccialli, F. (2021). Anomaly detection and trust authority in artificial intelligence and cloud computing. *Computer Networks*, *184*, 107647.
5. Kalaivani, K., & Chinnadurai, M. (2021). A hybrid deep learning intrusion detection model for fog computing environment. *Intelligent Automation and Soft Computing*, *30*(1), 1-15.
6. Elmasry, W., Akbulut, A., & Zaim, A. H. (2021). A design of an integrated cloud-based intrusion detection system with third party cloud service. *Open Computer Science*, *11*(1), 365-379.
7. Vadgaonkar, P. H. (2020). Network anomaly detection and user behavior analysis using machine learning. *International Journal of Computer Applications*, *175*(13), 47-53.
8. rao Gondi, S. (2020). Intelligent Fault Detection and Classification in Semiconductor Manufacturing Using Statistical Learning Techniques. Transactions on Smart Electrical and Computational Systems, 7, 26-31.
9. Shamshirband, S., Fathi, M., Chronopoulos, A. T., Montieri, A., Palumbo, F., & Pescapè, A. (2020). Computational intelligence intrusion detection techniques in mobile cloud computing

environments: Review, taxonomy, and open research issues. *Journal of Information Security and Applications*, *55*, 102582.

10. Aslan, Ö., Ozkan-Okay, M., & Gupta, D. (2021). Intelligent behavior-based malware detection system on cloud computing environment. *IEEE Access*, *9*, 83252-83271.