Real-Time Anomaly Detection System (RADS): A Lightweight, Database-Free Approach for System Performance and Security Monitoring

Nagma Khan[1], Shakila Siddavatam[2]

[1] Master Student, Department of Computer Science, Abeda Inamdar Senior College, India

[2] Head of Department, Department of Computer Science, Abeda Inamdar Senior College, India

Abstract

Modern computing systems require continuous monitoring to maintain stability, performance efficiency, and security. Sudden anomalies such as abnormal CPU utilization, excessive memory consumption, or unusual network activity may indicate performance degradation or malicious behavior. Traditional anomaly detection systems often depend on centralized databases, historical log analysis, or computationally intensive machine learning models, which introduce processing delays and increase system overhead.

This paper proposes a Real-Time Anomaly Detection System (RADS), a lightweight and database-free framework designed for instant detection of abnormal system behavior. The system continuously monitors CPU, RAM, and network usage using Python-based monitoring techniques and applies threshold-based and adaptive detection mechanisms. Upon anomaly detection, RADS immediately generates alerts through Discord, enabling rapid response without manual supervision. Experimental evaluation demonstrates that the proposed system achieves accurate real-time monitoring with minimal resource consumption, making it suitable for local systems, academic environments, and small-scale servers.

The proposed system is optimized for Ubuntu-based environments but can be extended to other operating systems with minimal changes. Experimental evaluation demonstrates that RADS delivers accurate, real-time anomaly detection with minimal resource overhead. By combining performance monitoring and security awareness in a single, accessible framework, RADS contributes a practical solution for real-time system reliability and protection

Keywords:

Real-time monitoring, anomaly detection, system performance, security monitoring, resource utilization.

1. Introduction

The rapid expansion of digital infrastructure and online services has significantly increased dependence on continuously operating computing systems. Modern software environments execute multiple processes simultaneously, resulting in dynamic fluctuations in processor utilization, memory consumption, and network traffic. These variations are often unpredictable and may arise from legitimate workload changes, software inefficiencies, or malicious activities. If abnormal system behavior is not detected at an early stage, it can lead to performance degradation, system instability, service interruption, and potential security breaches that affect overall system reliability.

A Real-Time Anomaly Detection System (RADS) is a monitoring framework designed to continuously observe system resource usage and immediately identify deviations from normal operational patterns. Unlike traditional monitoring approaches that rely on periodic log inspection or historical data analysis, RADS focuses on live system monitoring and instant anomaly reporting. The system operates without dependence on persistent database storage, thereby reducing computational overhead and enabling faster response to abnormal events. This real-time capability allows administrators or users to take corrective action before minor irregularities evolve into critical failures.

Existing anomaly detection methodologies employ statistical analysis, clustering techniques, and machine learning-based models to recognize abnormal patterns within system behavior [1]. While these approaches provide high detection accuracy, they typically require large training datasets, complex deployment infrastructures, and continuous model maintenance. Such requirements increase resource consumption and limit their effectiveness in lightweight environments such as personal systems, academic laboratories, or small-scale servers [2].

Furthermore, many modern monitoring platforms are primarily designed for enterprise or cloud-scale infrastructures. These systems depend heavily on centralized logging mechanisms and distributed analytics frameworks, which introduce latency between anomaly occurrence and detection. Although recent research on real-time anomaly detection systems has improved responsiveness,

most solutions remain focused on large cloud data centers rather than standalone or locally deployed systems [3]. As a result, there remains a need for efficient monitoring solutions that provide real-time detection without requiring complex infrastructure.

To address these limitations, this research proposes the Real-Time Anomaly Detection System (RADS), a lightweight and database-free monitoring framework capable of continuous system observation, instant alert generation, and efficient anomaly detection with minimal resource overhead. The proposed system integrates performance monitoring with security awareness, enabling early identification of abnormal conditions such as CPU overload, memory exhaustion, abnormal network activity, Distributed Denial of Service (DDoS) attacks, and unauthorized cryptomining operations. By emphasizing simplicity, responsiveness, and low computational cost, RADS provides a practical solution for maintaining system reliability and security in resource-constrained environments.

1.1. Problem Definition

System anomalies may occur due to workload spikes, software faults, configuration errors, or malicious cyber activities. Traditional monitoring systems rely on delayed log analysis or heavy backend infrastructure, preventing immediate response.

1.2. Significance of the Study

A lightweight real-time monitoring system enhances operational reliability by enabling early detection of abnormal behavior. RADS reduce administrative effort while improving system visibility and security awareness. The solution is particularly beneficial for personal systems, academic laboratories, and small organizational servers where enterprise monitoring tools are impractical.

1.3 Proposed Solution

The proposed RADS framework continuously collects system metrics and evaluates them using efficient anomaly detection mechanisms. The system processes data in memory and generates instant alerts without storing large datasets.

Key contributions include:

- Continuous real-time monitoring

- Lightweight database-free design
- Immediate anomaly alerting
- Low computational overhead
- Simple deployment for local environments

2. Literature Review

Anomaly detection has emerged as an essential research domain in system monitoring and cybersecurity, focusing on identifying unusual patterns that deviate from normal system behavior. Such deviations may represent performance failures, configuration errors, or malicious cyber activities. Over the years, researchers have proposed several approaches ranging from statistical analysis to machine learning-based detection methods.

Chandola et al. [1] presented one of the most comprehensive surveys on anomaly detection techniques, classifying approaches into statistical, proximity-based, clustering, and learning-based models. Their study emphasized that anomaly detection systems must balance detection accuracy with computational efficiency. Although machine learning approaches demonstrate strong performance, they often require labeled datasets, extensive training, and continuous model updates, which limit their practicality in real-time and resource-constrained environments.

Ahmed et al. [2] examined network anomaly detection techniques and highlighted challenges associated with real-world deployment. The authors noted that many monitoring frameworks depend on centralized data collection and heavy analytical processing, leading to increased latency in anomaly identification. Their findings indicate that real-time responsiveness remains a major challenge in traditional monitoring solutions.

With the growth of cloud computing infrastructures, researchers have explored large-scale anomaly detection mechanisms designed for distributed systems. Barbhuiya et al. [3] proposed a Real-Time Anomaly Detection System (RADS) for cloud data centers that analyzes telemetry data to detect abnormal operational conditions. While effective for enterprise-scale environments, such systems require complex infrastructure and are less suitable for standalone machines or academic laboratory setups.

Recent studies also emphasize system-level monitoring using lightweight tools. Practical monitoring solutions using scripting environments and operating system metrics have demonstrated improved efficiency for local deployments. Python-

based monitoring libraries such as psutil enable direct access to system resource statistics without requiring specialized hardware or database integration [5]. These approaches support continuous monitoring with minimal overhead, making them ideal for lightweight anomaly detection systems.

In addition, modern monitoring platforms increasingly integrate instant communication mechanisms to improve response time. Real-time notification systems, including webhook-based messaging services such as Discord, allow automated alert delivery across multiple devices, ensuring rapid awareness of abnormal events without manual supervision [6].

Despite significant progress, existing research largely focuses on either high-accuracy machine learning models or enterprise monitoring infrastructures. There remains limited research addressing lightweight, database-free, and real-time anomaly detection systems specifically tailored for local environments. Many solutions still rely on historical log storage or centralized processing, which introduces delays and increases resource consumption.

To address this research gap, the proposed Real-Time Anomaly Detection System (RADS) combines continuous system monitoring, lightweight threshold-based analysis, and instant alert generation within a database-free architecture. The system aims to provide efficient anomaly detection while maintaining simplicity, scalability, and minimal computational overhead.

3. Methodology (Development Process)

3.1 Research Design

This research adopts a design and development methodology focused on building a lightweight Real-Time Anomaly Detection System (RADS) for monitoring system performance and security. The system continuously observes CPU usage, memory utilization, and network activity to detect abnormal behavior in real time. The design integrates system monitoring techniques, anomaly detection mechanisms, and instant alert generation to ensure efficient and fast response without database dependency.

3.2 Information Gathering

• Secondary Data:
Research papers, IEEE journals, and articles related to anomaly detection, system monitoring, and cybersecurity were reviewed to understand existing approaches and their limitations.

• Technical Research:
A study of Python-based monitoring tools, Linux environments, and real-time notification services was conducted to select suitable technologies ensuring lightweight operation and real-time performance monitoring.

3.3 System Architecture

The RADS is developed using a modular architecture to ensure efficient monitoring and data processing. The system integrates monitoring services with anomaly detection and notification modules to provide real-time analysis and instant alerts.

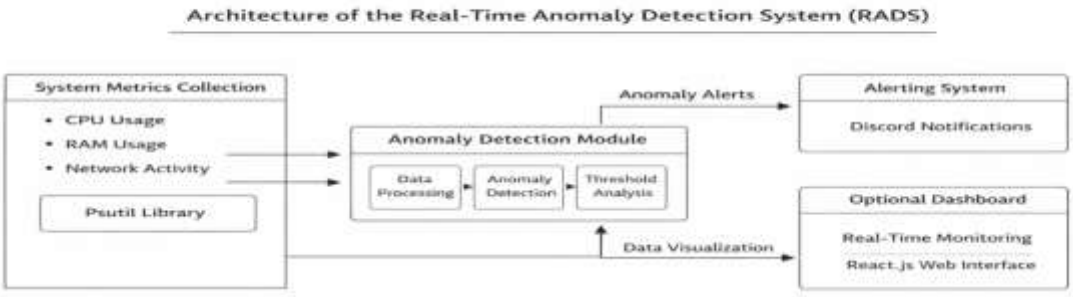

Fig 1: Architecture of the Real-Time Anomaly Detection System (RADS)

As illustrated in Fig. 1, the Monitoring Module collects CPU, RAM, and network statistics using the psutil library. The Anomaly Detection Module evaluates these metrics against predefined thresholds. When an anomaly is detected, the Notification Module sends an alert to a Discord channel. The Visualization Module optionally displays real-time metrics through a React.js dashboard.

4. Design and Implementation

4.1 User Interface

The RADS architecture is designed using a lightweight modular structure to support real-time monitoring and anomaly detection. The system consists of the following components:

• **Monitoring Layer :**
Implemented using Python to continuously collect system metrics such as CPU usage, memory utilization, and network activity.

• **Detection Layer :**
Applies threshold-based and adaptive anomaly detection techniques to identify abnormal system behavior in real time.

• **Notification Layer :**
Generates instant alerts using Discord webhooks whenever abnormal resource usage is detected.

• **Visualization Layer :**
An optional React.js dashboard provides real-time visualization of system performance metrics.

4.2 Technologies Used

Component	**Technology Used**
Programming Language	Python 3.10
Monitoring Library	psutil
Frontend Dashboard	React.js
Operating System	Ubuntu Linux
Detection Technique	Threshold-Based Detection
Notifications	Discord Webhooks

4.3 User Interface (UI) Screenshots

The Real-Time Anomaly Detection System (RADS) provides a simple and user-friendly interface for monitoring system performance. The dashboard displays live system metrics and enables users to observe resource usage continuously.

The interface is designed to provide clear visualization and instant awareness of abnormal system activity.

4.3.1 Monitoring Dashboard

The monitoring dashboard displays real-time information related to:

- CPU utilization
- Memory usage
- Network activity

Users can easily monitor system health through graphical visualization updated in real time.

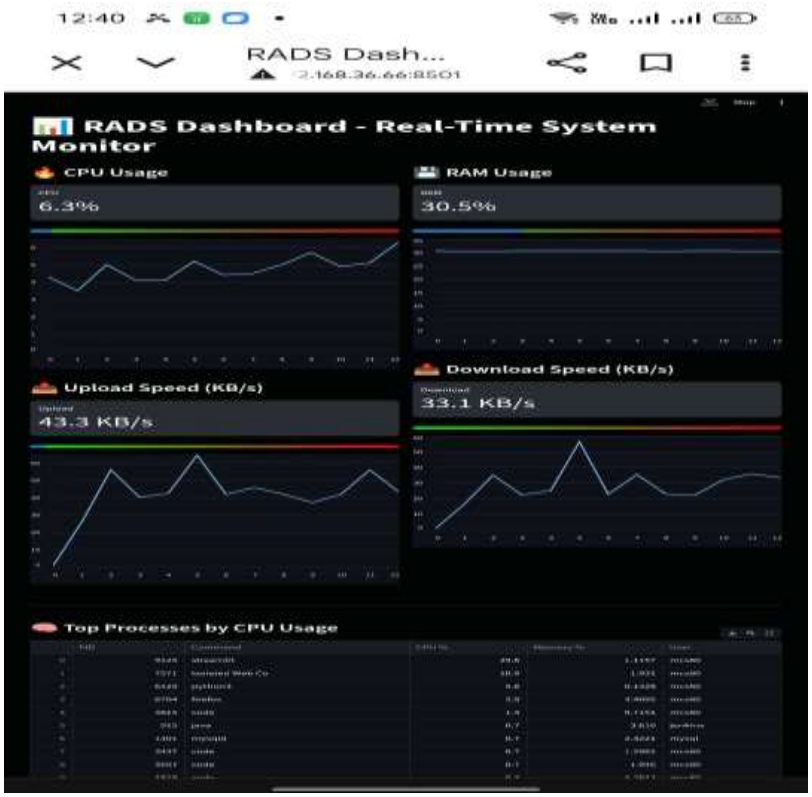

Fig 2: Real-Time Resource Monitoring Dashboard on mobile

4.3.2 Discord Alert Notification

When an anomaly is detected, the system automatically sends an alert message containing anomaly details and timestamp information to a Discord channel. This enables immediate notification across devices without manual monitoring.

Fig. 3: Discord-Based Anomaly Alert Notification

The dashboard visualization improves interpretability, while Discord notifications ensure timely awareness across devices.

5. Test Case Description

Test Case 1: System Monitoring Initialization
Verifies successful collection of CPU, memory, and network metrics.

Test Case 2: CPU Usage Anomaly Detection
System correctly identifies abnormal processor utilization exceeding threshold limits.

Test Case 3: Memory Overload Detection
Detects excessive memory consumption indicating potential resource leaks.

Test Case 4: Network Activity Monitoring
Identifies abnormal traffic patterns suggesting suspicious behavior.

Test Case 5: Alert Notification Mechanism
Ensures instant Discord notification upon anomaly detection.

Test Case 6: Continuous Monitoring Stability
Confirms low system overhead during long-term operation.

6. Discussion

6.1 Strengths of the System

- Lightweight and database-free architecture
- Immediate anomaly detection and alerting
- Minimal computational overhead
- Easy deployment in local environments
- Integration of performance monitoring and security awareness

6.2 Challenges and Limitations

- Threshold values may require tuning for different systems
- Detection accuracy depends on proper configuration
- Currently optimized mainly for Ubuntu environments

6.3 Future Scope

Future enhancements may include:

- Machine learning-based predictive anomaly detection
- Multi-device monitoring capability
- Cross-platform compatibility
- Integration with email and enterprise messaging alerts
- Automated response actions for critical anomalies

7. Conclusion

This research presented the Real-Time Anomaly Detection System (RADS), a lightweight framework designed for efficient monitoring of system performance and security conditions. By eliminating database dependency and emphasizing instant alerting, RADS provides rapid anomaly detection with minimal resource consumption. The system demonstrates that effective real-time monitoring can be achieved without complex infrastructure, making it suitable for academic, personal, and small-scale operational environments.

8. References

[1] V. Chandola, A. Banerjee, and V. Kumar,
"Anomaly detection: A survey,"
ACM Computing Surveys, vol. 41, no. 3,
pp. 1–58, 2009.

[2] M. Ahmed, A. N. Mahmood, and J. Hu,
"A survey of network anomaly detection techniques,"
Journal of Network and Computer Applications,
vol. 60, pp. 19–31, 2016.

[3] S. Barbhuiya, A. Papadopoulos,
J. Eliasson, and P. Tsigas,
"RADS: Real-time anomaly detection system for
cloud data centres,"
arXiv:1811.04481, 2018.

[4] Python Software Foundation,
Python 3 Documentation.
[Online]. Available: https://docs.python.org/3/

[5] Psutil Development Team,

Psutil Documentation.

[Online]. Available: https://psutil.readthedocs.io/

[6] Discord Inc.,

Discord Developer Documentation.

[Online]. Available: https://discordpy.readthedocs.io/