

# SNMP Network Monitoring and Cyber Alert System

Wan Mohamad Hariz Bin Wan Marzuki<sup>a</sup>, Adamu Abubakar Ibrahim<sup>a</sup>

<sup>a</sup>Department of Computer Science, KICT, International Islamic University Malaysia, Kuala Lumpur, Malaysia

## ARTICLE INFO

**Keywords:**  
alert system  
FastAPI  
network management  
network monitoring  
Next.js  
SNMP

## ABSTRACT

This project addresses the critical need for accessible, low-cost, and easily deployable network monitoring systems, particularly for smaller organizations and educational institutions that often face budget limitations and a lack of technical expertise. Existing network monitoring solutions frequently involve complex implementations or high costs, hindering immediate deployment by these entities. To bridge this gap, an SNMP-based network monitoring and alert system is proposed. This system aims to streamline the monitoring process by periodically polling network devices for key performance metrics such as CPU utilization and interface bandwidth usage and automatically triggering email alerts when predefined thresholds are exceeded. The application is developed using FastAPI for the backend and Next.js for the frontend to ensure high performance. Designed for deployment on a Linux server, the system offers a simplified and cost-effective solution for continuous network surveillance and prompt anomaly detection.

## 1. Introduction

In today's era of technology, the performance and reliability of computer networks are critical to society. Lee (2014) emphasizes that network monitoring and measurement have become increasingly important in modern complicated networks. However, many of the existing solutions for network monitoring reveal major challenges for implementation. Kocak and Ergul (2014) note that developing effective network monitoring tools requires addressing various technical complexities that can be overwhelming for organizations with limited technical expertise.

Small organizations and educational institutions often find themselves in a bind as they are constrained by budget limitations and lack technical expertise. To address this gap, a simplified approach can be proposed with an SNMP network monitoring and alert system. SNMP is a widely used protocol that facilitates monitoring and managing network devices by collecting real-time data through performance queries. To complement the monitoring system, an alert system automatically processes the data and triggers an alert based on predefined thresholds when certain metrics exceed the limit. The combination of these systems helps streamline the monitoring process and ensures that fast action can be taken when an anomaly happens, making it a viable and cost-effective solution.

The core problem identified in current literature is the lack of accessible, low-cost, and easy-to-deploy network

monitoring systems for continuous performance monitoring. Sultana and Jairam (2019) highlight that existing systems often involve complex implementations or high costs that prevent immediate deployment by smaller organizations. This creates a significant gap where organizations require automated monitoring of essential network metrics via SNMP but lack access to appropriate solutions that can be verified within simulated environments such as GNS3.

The primary objective of this project is to design and implement a basic SNMP-based network monitoring and alert system. The system polls critical network devices for key metrics such as CPU utilization and interface bandwidth usage. Collected data are stored in a structured SQLite relational database to enable efficient querying and data analysis. Furthermore, the system is designed to trigger email alerts automatically when predefined thresholds are exceeded.

## 2. Related Work

Several open-source network monitoring solutions exist, including Zabbix (Zabbix, n.d.), Nagios Core (Nagios, n.d.), and LibreNMS (LibreNMS, n.d.). These platforms offer robust features such as auto-discovery, extensive plugin support, and multi-channel alerting. However, these enterprise-grade systems often present significant barriers for small organizations. Findings indicate that complex configuration and high resource requirements lead to a high abandonment rate among non-technical users (Maulana and Al-Khowarizmi, 2021; Arcia-Moret et al., 2021). Furthermore, the maintenance overhead and steep learning curve required

 hariz.marzuki@live.iium.edu.my (W.M.H.B.W. Marzuki);  
hariz.marzuki@live.iium.edu.my (A.A. Ibrahim)  
ORCID(s):

**Table 1**

Comparison of existing monitoring solutions.

| Feature        | Zabbix        | Nagios Core  | LibreNMS      |
|----------------|---------------|--------------|---------------|
| Deployment     | High          | High         | Medium        |
| Learning curve | Steep         | Steep        | Moderate      |
| Auto-discovery | Advanced      | Plugin-based | Excellent     |
| Web interface  | Modern        | Basic        | Modern        |
| Alerting       | Multi-channel | Plugin-based | Multi-channel |

for these systems make them less suitable for organizations with limited resources.

Several empirical research studies are associated with SNMP. Baikole et al. (2026) demonstrated that implementing SNMP using the Rapid Application Development methodology effectively improves network monitoring and facilitates faster restoration of device access. The system enabled administrators to detect faults quickly, manage network devices efficiently, and reduce downtime in network environments. Khan et al. (2026) proposed a trusted SNMP framework designed to ensure integrity evaluation within distributed network environments. The results showed that the enhanced protocol improves reliability and security by detecting unauthorized modifications in network data exchanges, thereby strengthening monitoring accuracy and trust in distributed network management systems.

Carranza et al. (2026) experimentally evaluated the integration of the Open Shortest Path First routing protocol with SNMP for network monitoring. Findings revealed that the integration improves network availability, enhances fault detection, and supports high availability by enabling real-time monitoring of routing performance and network device status. Zikra et al. (2026) implemented an SNMP-based monitoring system using the Cacti application with Telegram notification support on Ubuntu Linux. The results indicated that administrators could monitor network performance in real time and receive instant alerts about device failures, improving response time and overall network management efficiency. Safrianti et al. (2021) developed a real-time network monitoring model based on SNMP to observe device status and network traffic conditions. The results showed that the system successfully provided continuous monitoring and rapid fault detection, enabling network administrators to maintain stability and respond promptly to network disturbances.

### 3. Methodology

#### 3.1. Development Approach

The project employs the Agile software development methodology to guide the creation of the system. This approach highlights collaborative, iterative development cycles, known as sprints, where working software is delivered incrementally. Each sprint involves structured phases: planning, design, implementation, testing, and deployment. This repeatable cycle enabled ongoing refinement of the system,

incorporating user feedback and operational requirements with each iteration to ensure defects are detected early.

#### 3.2. Requirements Specification

Requirements were gathered to ensure system functionality and operational expectations were satisfied.

- **Automated device discovery:** The system must automatically discover network devices and their interfaces to ensure all relevant network elements are monitored.
- **Comprehensive monitoring:** It must provide monitoring of key device metrics including availability, CPU utilization, and network bandwidth consumption using SNMP.
- **Intelligent alerting:** The system must allow for the configuration of customizable thresholds and provide timely notifications via email when these thresholds are breached.
- **Non-functional requirements:** The system requires an intuitive web-based interface for usability and must ensure high availability to support continuous network surveillance.

#### 3.3. System Architecture

The application is developed using FastAPI for the backend and Next.js for the frontend to ensure high performance. The system is developed with a modern three-tier architecture that separates the presentation, application logic, and data management. The system comprises a React-based frontend (presentation tier), a FastAPI backend (application tier), and a SQLite database (data tier).

The backend manages alerting logic, handles authentication, processes SNMP data, and exposes RESTful API endpoints. FastAPI was chosen due to its native support for asynchronous operations, which is critical for efficient SNMP polling. The `pysnmp` library is utilized to facilitate monitoring and managing network devices by collecting real-time data.

#### 3.4. Logical Design

The logical design models the interactions within the system. As shown in Fig. 2, the Network Admin is the main user who interacts with the system to perform monitoring and management tasks. The system communicates with

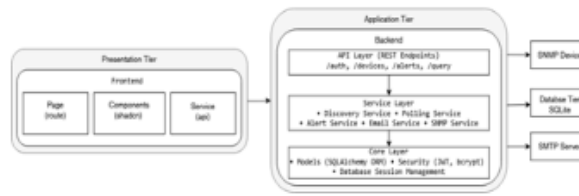


Figure 1: Three-tier architecture of the SNMP monitoring system.

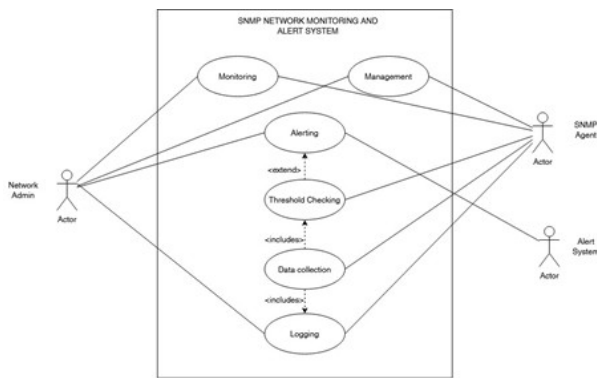


Figure 2: Use case diagram illustrating actor-system interactions.

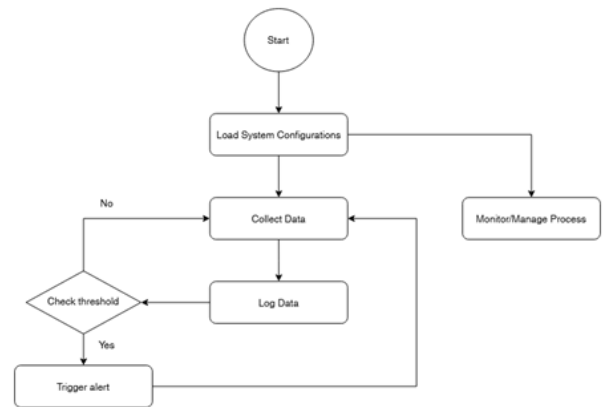


Figure 3: Operational flowchart of the monitoring system.

SNMP Agents on network devices to perform core functions such as data collection, which includes logging all relevant events.

The system operation follows a continuous loop mechanism. As illustrated in Fig. 3, the process begins by loading necessary system configurations. The system then enters a continuous loop where it collects data, logs it, and proceeds to check thresholds. If a threshold condition is met, the system triggers an alert before returning to the data collection step.

### 3.5. User Interface Design

The prototype for this project was developed with Figma to showcase key features through intuitive, high-fidelity screens.

#### 3.5.1. Dashboard Overview

Figure 4 presents the wireframe for the dashboard, which serves as an overview of the network monitoring center. It features a filter panel on the left where administrators can narrow down the network address range, and a central display for key metrics such as CPU utilization. A timeline panel on the right tracks system activity and alert triggers.



Figure 4: Wireframe design of the main dashboard interface.

#### 3.5.2. Device Management Interface

The Devices page, shown in Fig. 5, is designed to list all monitored devices. At the top, a service-availability summary displays percentages of devices that are available or down, alongside a line chart tracking packet counts over time. The bulk of the screen is devoted to a table detailing each device's hostname, IP address, uptime, and location.

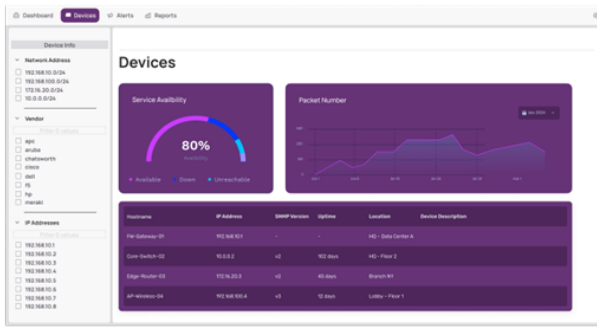


Figure 5: Wireframe of the devices list and availability summary.

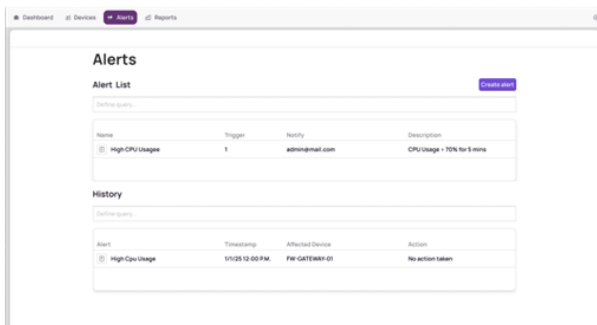


Figure 6: Wireframe of the alert configuration and history page.

### 3.5.3. Alert System Interface

The Alerts tab reveals the health-rules management interface. As illustrated in Fig. 6, the main content is a table of all active alert definitions, where each row shows the trigger condition and notification targets. Beneath this list, a history section logs past alert occurrences with timestamps and actions taken.

## 3.6. Experimental Setup

The network architecture used to evaluate the system was implemented in a GNS3 simulation environment, allowing controlled testing of SNMP operation without relying on physical hardware. The topology, shown in Fig. 7, consists of two Cisco routers (R1 and R3), three generic Layer 2 switches, and four host terminals. SNMPv2 with community string FYP was configured on the routers as they serve as the primary SNMP-enabled devices for data collection.

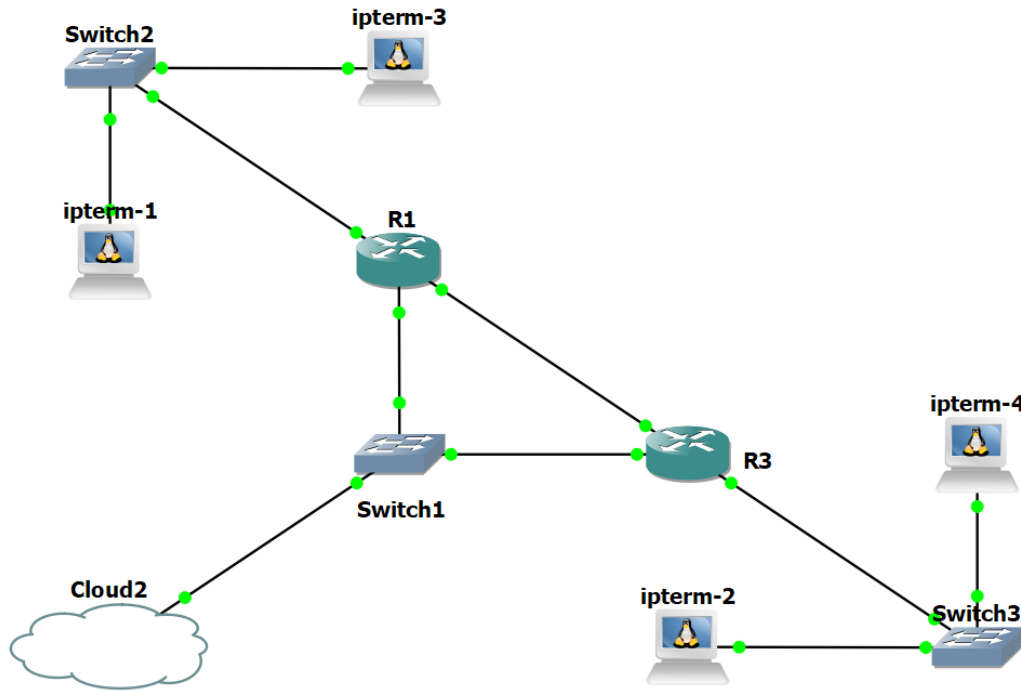


Figure 7: GNS3-simulated network topology used for system validation.

## 4. Results and Discussion

### 4.1. System Output

The completed system provides automated network discovery, continuous monitoring, and a threshold-based alerting system with email notification in place. Developed with a modern three-tier architecture, the system provides a comprehensive monitoring experience for the user.

#### 4.1.1. Dashboard Interface

The main dashboard serves as the primary interface and provides an overview of the network status. As shown in Fig. 8, it displays the number of monitored devices, their current online/offline status, and summarizes recent alert activity. The dashboard also displays key SNMP metrics such as CPU load, memory usage, and network traffic trends through interactive charts. This allows users to quickly identify abnormal patterns or performance degradation.

#### 4.1.2. Device Performance Visualization

The Device Details page visualizes historical and SNMP metrics for specific devices. As illustrated in Fig. 9, these include CPU utilization, memory usage, inbound/outbound bandwidth, and interface statistics. Graphs are generated using the Recharts library and updated continuously based on the polling interval configured during setup.

#### 4.1.3. Alert Management

The Alert page displays all triggered alerts generated when device metrics exceed user-defined thresholds. Shown in Fig. 10, each alert contains the device name, timestamp, affected metric, threshold value, and severity level. This

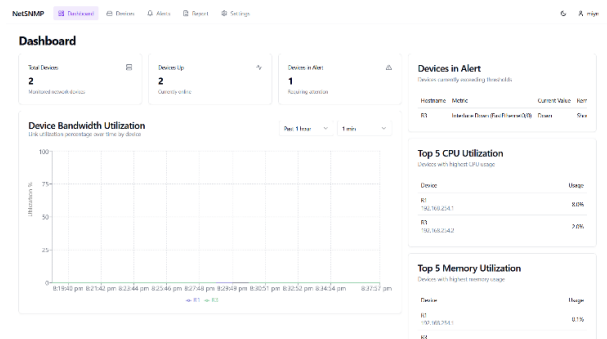


Figure 8: Dashboard page displaying network status and key utilization metrics.

interface enables quick identification of critical issues and supports troubleshooting activities by maintaining a historical log of anomaly events.

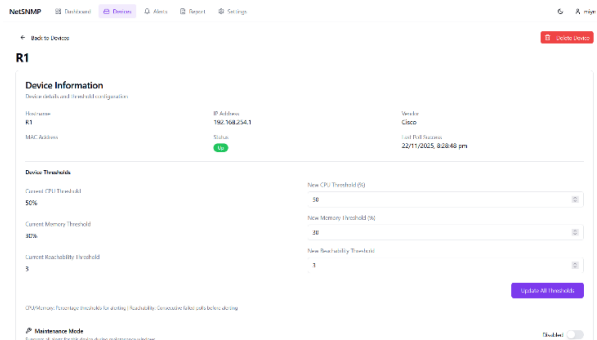
### 4.2. System Testing

Testing focused on validating system functionality, verifying accurate SNMP data retrieval, and ensuring the reliability of the alert system. A structured test plan was developed and executed based on black-box testing methodology.

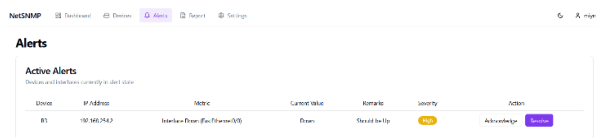
As summarized in Table 2, the system exhibits a high level of functional reliability, stability, and readiness for operational use. Testing confirmed that the polling engine accurately retrieves SNMP metrics at the configured intervals and dynamically adapts to changes in system settings without requiring a restart. Furthermore, threshold-based alerts are generated, logged, and timestamped correctly.

**Table 2**  
System validation results.

| Test case | Description                                  | Status |
|-----------|----------------------------------------------|--------|
| TC01      | Start device discovery for selected IP range | Pass   |
| TC02      | Poll device every 30 seconds                 | Pass   |
| TC03      | Exceed CPU threshold to trigger alert        | Pass   |
| TC04      | Change polling interval in settings          | Pass   |
| TC05      | View device details charts (real-time)       | Ready  |
| TC06      | View active alert timestamp                  | Pass   |
| TC07      | Handle invalid SNMP community string         | Pass   |



**Figure 9:** Performance metrics visualizing historical utilization trends.



**Figure 10:** Alert display showing threshold violations and severity levels.

## 5. Conclusions

This project successfully exhibits a working network monitoring system capable of receiving SNMP data from simulated devices and providing meaningful metrics through a modern web interface. Functionally, the system monitors network devices using SNMP, collects important performance metrics, and triggers email notifications when threshold conditions are met. The interface with Gmail SMTP allows the system to provide alarm notifications efficiently without the need to establish a specialized email service.

The project is subject to several limitations. To make development and integration easier, SNMP version 2c was used for this solution. However, SNMPv2c lacks robust security protection and transmits community strings in plain text. This constraint would need to be fixed before implementing the system in real-world settings. Furthermore, the system only uses virtualized devices in GNS3, which might not accurately reflect the configuration differences or performance traits of real network gear.

Future iterations of the system can include several improvements to boost usability, security, and usefulness.

Upgrading from SNMPv2c to SNMPv3 would considerably increase authentication and data protection. To provide enterprise-level alerting, the system might allow extra notification channels such as SMS, Telegram bots, or web-hook interfaces. Additionally, migrating the backend to a production-ready server such as Gunicorn with Nginx would improve scalability. The system accomplishes its main goals and offers a useful basis for future development even if it operates in simulated and limited contexts.

## References

- Arcia-Moret, A., Gomez, J., Sathiaselalan, A., 2021. Octopus: A zero-cost architecture for stream network monitoring. arXiv preprint arXiv:2105.01183.
- Baikole, A., Purwadi, J., Indriyanta, G., 2026. Evaluating SNMP protocol for monitoring and restoring device access using RAD method. International Journal of Informatics and Computation 8, 146–157.
- Carranza, H.J.S., Riera, J.J.A., Morocho, B.L.C., Wong, P.M.W., 2026. Integration of OSPF and SNMP for network monitoring and high availability: Experimental evaluation. InGenio Journal 9, 87–112.
- Khan, H.M.A., Sarwar, N., Hamdi, M., Inayat, U., 2026. Trusted Simple Network Management Protocol for integrity evaluation in distributed environment. Journal of Network and Systems Management 34, 48.
- Kocak, H., Ergul, S., 2014. A step on developing network monitoring tools. International Journal of Computer Applications 99, 37–42.
- Lee, S., 2014. Network monitoring: Present and future. Computer Networks doi:10.1016/j.comnet.2014.03.007. accessed: 2025-04-09.
- LibreNMS, n.d. Features – librenms docs. URL: <https://docs.librenms.org/Support/Features/>. accessed: 2025-04-13.
- Maulana, H., Al-Khowarizmi, A., 2021. Analyze and designing low-cost network monitoring system using icinga and raspberry pi. IOP Conference Series: Earth and Environmental Science 704, 012038.
- Nagios, n.d. Features | nagios open source. URL: <https://www.nagios.org/about/features/>. accessed: 2025-04-13.
- Safianti, E., Sari, L.O., Sari, N.A., 2021. Real-time network device monitoring system with Simple Network Management Protocol (SNMP) model, in: Proceedings of the 3rd International Conference on Research and Academic Community Services (ICRACOS), pp. 122–127.
- Sultana, A., Jairam, G.B., 2019. A review of applications and approaches of network monitoring. International Journal of Innovative Research in Computer Science & Technology (IJIRCST) 7. URL: <https://ssrn.com/abstract=3509478>.
- Zabbix, n.d. Zabbix features overview. URL: <https://www.zabbix.com/features>. accessed: 2025-04-13.
- Zikra, F., Anwar, A., Safar, I., 2026. Monitoring networks based on Simple Network Management Protocol with Cacti application and Telegram notifications on Linux Ubuntu. Jurnal Teknologi Rekayasa Informasi dan Komputer 9, 55–65.