SSRG International Journal of Computer Science and Engineering
ISSN: 2348–8387 / https://doi.org/10.14445/23488387/IJCSE-V12I15P104

Volume 12 Issue 5, 29-34, May 2025

Original Article

Developing an Attack Warning Feature for Open Source Code

Thi Ngoc Linh Tran

Faculty of Electronics Engineering, Thai Nguyen University of Technology, Thai Nguyen City, Vietnam.

Corresponding Author : tranngoclinh@tnut.edu.vn

Received: 17 March 2025 Revised: 25 April 2025 Accepted: 13 May 2025 Published: 30 May 2025

Abstract *- Network security is a major concern in today's digital era. Computer networks are constantly exposed to risks posed by hackers. To enhance network security, it is essential to have effective solutions that can scan for and detect signs of potential attacks. This study focuses on developing an attack alert feature using the open-source platform Nagios, one of the most widely used network monitoring tools. One of Nagios' key advantages is its open-source nature, which allows users to easily customize, edit, and add new features. The attack detection feature implemented in this project is based on an anomaly detection algorithm for TCP connection-oriented protocols. The algorithm is simple, easy to install, and capable of effectively detecting large-scale attacks using multiple techniques. When abnormal behavior is detected, Nagios sends an alert to the system administrator. With these capabilities, administrators can respond more proactively to potential threats and address issues at the early stages of an attack, minimizing damage and maintaining system integrity.*

Keywords – *Attack Warning Fearture, Open Source Code, Nagios, Network Security, Network Monitoring Tool.*

1. Introduction

Cybersecurity is also extremely important in today's world. With the rapid development of computer networks, users have gained numerous benefits. However, alongside these advantages come dangers from hackers. Network administrators constantly find themselves in a passive position when facing these threats. They are unable to predict the timing or method of attacks, so they often cannot respond in a timely manner. This leads to system disruptions, affects services, and prevents users from accessing them. One of the most common forms of attack today is the Denial-of-Service (DoS) attack on websites. This type of attack comes in many forms, causes significant damage and loss, and is difficult to detect, making it a powerful weapon for hackers [1].

The "flood" attack method, which generates a large number of requests, can slow down the system or even cause it to become disrupted. Unlike destructive attacks using viruses, DDoS (Distributed Denial of Service) attacks utilize a massive number of computers (potentially up to millions) to send HTTP GET messages that overwhelm the system, making it difficult to analyze using network administration tools. Among all types of DDoS attacks, more than 90% exploit the TCP transport protocol. A common and effective form of attack takes advantage of the TCP three-way handshake mechanism [2]. In this method, connections are not fully established, yet the server is forced to maintain these half-open connections, which consume resources [3]. When a SYN packet is received, the server responds with a SYN/ACK packet at the corresponding service port. Until the server receives the final ACK packet from the client, it must hold the half-open connection for a timeout period of approximately 75 seconds. The server stores these half-open connections in memory until the memory is full, at which point it starts dropping them [4].

To effectively manage resources and protect the network system from unpredictable threats, it is necessary to have a reliable and accurate resource performance evaluation and monitoring system. With such a system, administrators can assess server demand, network device usage, and security status, leading to better, more efficient, and safer resource utilization strategies that are also cost-effective. Therefore, a monitoring solution that provides an accurate overview of the system is essential. A tool with the capability to alert administrators immediately when signs of an attack are detected is crucial. This feature helps system administrators become more proactive in protecting the network, allowing them to prepare appropriate countermeasures and thereby enhance the system's ability to deliver services to users [5].

Currently, numerous tools and software solutions are available for network system monitoring. Choosing a suitable tool to install on a server—used for observing and storing information from other servers and network devices—is an important consideration. With the help of such tools, network

management becomes much simpler [6]. For example, administrators can easily define policies for each server, service, device, or user within the monitoring tool and configure alerts at different levels for the administrators. From a remote location, administrators can view alerts and take appropriate measures to address and resolve issues indicated by those warnings. This includes decisions about resource replacement based on current conditions, understanding how applications use system resources, identifying types of attacks on the system, and determining corresponding countermeasures. As a result, administrators can make appropriate configuration recommendations for servers running software or services, as well as for security devices that are necessary [7].

The author studies Nagios, an open-source tool that is highly effective for monitoring network systems. It is suitable for both large and small network environments. The biggest advantage of Nagios is its flexibility and extensibility—it can be used in various ways, and its features can be expanded according to user needs through the development of plugins. This study will implement an attack alert feature to be integrated into Nagios, helping to detect and promptly warn when the system shows signs of being attacked.

Thanks to this feature, issues within the system can be addressed in the most efficient manner possible. The primary purpose of Nagios is to monitor the entire IT infrastructure, ensuring that systems, applications, security, services, and processes are functioning correctly. In the event of any abnormal signs, Nagios will send alerts to administrators, allowing them to begin the recovery process before an incident occurs and affects operations or users. The Nagios tool helps accurately identify issues affecting the critical infrastructure of the network system.

2. Theoretical Framework

Nagios is known as an open-source program for monitoring networks, services, and servers [8]. In essence, this tool serves as a framework for observing devices, allowing administrators to quickly compile various command lines into configurations to collect information. There are many external tools that support Nagios, and it is easy to integrate Nagios with other monitoring tools, such as NRPE and MRTG [9].

First, it is essential to gather the core information surrounding the general configuration of Nagios, so it is necessary to start with basic configuration files related to four key files: hosts, host groups, contacts, and services [9]. The default configuration files come with clear descriptions of the specific functions within each file. These files are located in Nagios's default installation directory: /usr/local/nagios/etc.

Nagios's configuration is very straightforward [10]. Servers running the same service can be grouped together, allowing administrators to easily track them in the Nagios web interface. Similarly, when multiple administrators manage different services, they can be grouped into contact groups. Suppose a server running the Nagios program is shut down or loses connection to a running service. In that case, Nagios will notify the responsible administrator or administrator group that manages that server or service.

Nagios runs as a monitoring service, checking servers and services. At the same time, external supporting tools are designated to collect information and send it back to the Nagios server at predefined scheduled intervals. When signs of issues are detected, Nagios alerts the system administrator via email or a message. All status information over time, historical logs, and report tables are uploaded to the web interface for detailed viewing.

The Nagios application runs on a central server, typically on a Linux or Unix system [11]. Each monitored server must run a Nagios-compatible service to communicate with the central server. The configuration files on the central server are executed to perform the necessary checks on remote servers and retrieve the collected information back to the central server. While the Nagios application must run on a Linux or Unix server, the remote servers can be any type of hardware or operating system as long as they are capable of communicating with the Nagios host.

Depending on the responses from the remote servers, Nagios will react accordingly based on its configuration settings. Depending on the type of remote check being performed, Nagios may execute checks using the capabilities of the remote server itself (such as checking if a file exists), or it may run a specialized check program (known as a Nagios plugin) to handle more complex checks (such as verifying the version of a specific software package). If the returned value is incorrect, Nagios will escalate the alert level using predefined and configured notification methods.

The NRPE tool is designed to allow users to execute Nagios plugins on remote Linux/Unix servers. The main reason for this is to enable Nagios to monitor resources on those servers (such as CPU usage, memory usage, etc.) from a distance. Since these resources are not usually exposed externally, a background program (called an agent) like NRPE must be installed on the remote Linux/Unix servers.

Additionally, Nagios can execute plugins on remote Linux/Unix servers via SSH, using a plugin called check_by_ssh that enables this functionality. Using SSH is more secure than NRPE, but it consumes significantly more CPU resources on both the monitoring server and the remote server. This becomes a concern when monitoring hundreds or thousands of servers. Therefore, using NRPE is generally a better option due to its lower CPU overhead.

The NRPE tool consists of two components: the check_nrpe plugin, located on the monitoring server, and the NRPE daemon, running on the remote Linux/Unix server. When Nagios needs to monitor the resources or service on a remote Linux/Unix server, Nagios executes the check_nrpe plugin, specifying which service or resource to check on the remote server. The check_nrpe plugin communicates with the NRPE daemon on the remote server through a secure connection protected by the SSL protocol. The NRPE daemon runs the appropriate Nagios plugin to check the specified service or resource as defined in its configuration. These Nagios plugins must already be installed on the remote server. Without them, the NRPE daemon cannot monitor anything. The NRPE daemon returns the result of the check to the check_nrpe plugin, which then passes it back to the Nagios software for processing. Nagios operations can be set up to perform checks in two ways: using direct checks and indirect checks. Direct check is the simplest method using NRPE to monitor specific local resources on the remote server. For example, CPU load, RAM usage, disk space, etc. Nagios can use NRPE to perform indirect checks on public services and resources of remote servers that the Nagios server cannot directly access. For example, suppose a remote server has the NRPE daemon and Nagios plugins installed and is able to communicate with another remote server (whereas the Nagios server itself cannot). In that case, the administrator can configure the NRPE service to allow the Nagios server to indirectly monitor that second remote server. In this setup, the NRPE service essentially functions as an intermediary service.

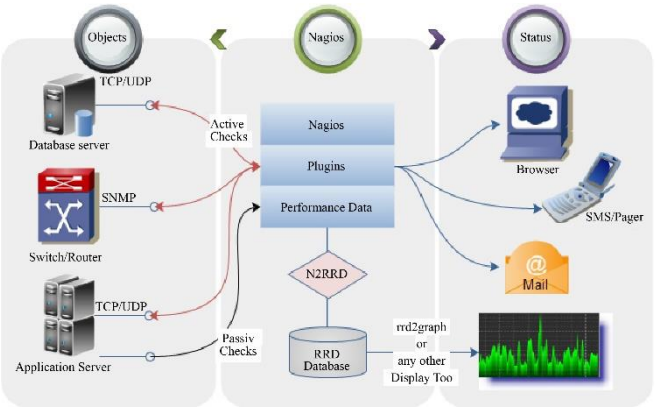

Fig. 1 Architecture of Nagios

3. Materials and Methods

3.1. Building an Attack Alert Plugin for Nagios

This feature is designed to detect early signs of attacks, whether carried out individually or in combination, that hackers use to avoid detection. The goal of developing this feature is to integrate it into Nagios Core to promptly detect Denial of Service (DoS) attacks on servers. With this functionality, system administrators can modify and customize it over time to suit the current network conditions. Moreover, it allows for flexible updates to match attack indicators corresponding to the methods used by hackers.

To promptly detect sudden large-scale attacks from the very beginning, it's essential to monitor and collect abnormal signs during the connection establishment phase—the first step in communication using the connection-oriented TCP protocol. In this phase, a client sends TCP_SYN messages to request a connection with the server. After receiving this message, the server responds with a TCP_SYN/ACK and allocates resources to maintain a half-open connection for about 75 seconds, waiting for a TCP_ACK response from the client. If the client never sends back a TCP_ACK, the server wastes resources maintaining that half-open connection.

Furthermore, if the number of TCP_SYN messages spikes due to many clients participating in this process, server resources can be exhausted very quickly. Hackers exploit this principle through various techniques, often combining multiple methods to avoid detection and complicate the defense and recovery process.

3.2. Detecting Signs of Denial-of-Service (DoS) Attacks

A Denial-of-Service (DoS) attack is a method of drastically increasing network traffic (bandwidth usage) by sending a large number of service connection requests to a server, causing the server to become overwhelmed and unable to respond, leading to system service disruptions or complete failure. DoS attacks remain a serious threat today. Although several countermeasures exist, most are either ineffective or only moderately effective.

This attack method is based on the TCP three-way handshake mechanism between a client and a server. One common attack based on this principle is the SYN flood attack. The attacker sends a large number of SYN requests with spoofed (non-existent) source IP addresses. These SYN requests are technically valid, so the target server responds with a SYN/ACK and allocates resources for a half-open connection. Since the spoofed source never replies with an ACK, the server ends up maintaining a large number of these half-open connections [7]. Preventing this kind of attack requires systems to detect early indicators at the very beginning of the attack. The detection method is usually based on anomaly detection algorithms.

Several anomaly detection methods consider various parameters, such as the CUSUM (Cumulative Sum Control Chart) combined with average cumulative thresholds, as well as the frequency of message pairs during data transmission (e.g., SYN–FIN pairs, SYN-ACK pairs). However, these methods are typically applied on border routers. For a system to detect DoS attacks effectively, a packet filter must be implemented. The success and accuracy of early detection depend heavily on the quality of this packet filter.

One detection algorithm designed for this purpose is the Adaptive Threshold Algorithm. This algorithm is simple, easy to implement, and suitable for Linux environments. It

measures the volume of TCP-SYN packets during the initial phase of communication (connection setup) and compares it against a predefined threshold. This is what distinguishes it from other anomaly detection methods. The threshold is determined over a fixed period based on an estimated average number of TCP-SYN packets. If the observed traffic exceeds this threshold, it is identified as an anomaly, and a warning is triggered.

Assume at time t:

xt: number of SYN packets received during time interval t

μ_{t-1}: average rate of SYN packets measured before time t

Alert condition is defined as:

If x_t ≥ (α + 1)·μ_{t-1}, then an alert is triggered at time t.

Where:

μ_t: average threshold value calculated over several sample intervals

α > 0: the percentage over the average rate, derived from sample intervals where thresholds were previously exceeded

This forms the basis of the adaptive threshold algorithm, which is used to detect signs of SYN flood attacks by comparing the current rate of SYN packets to a dynamic.

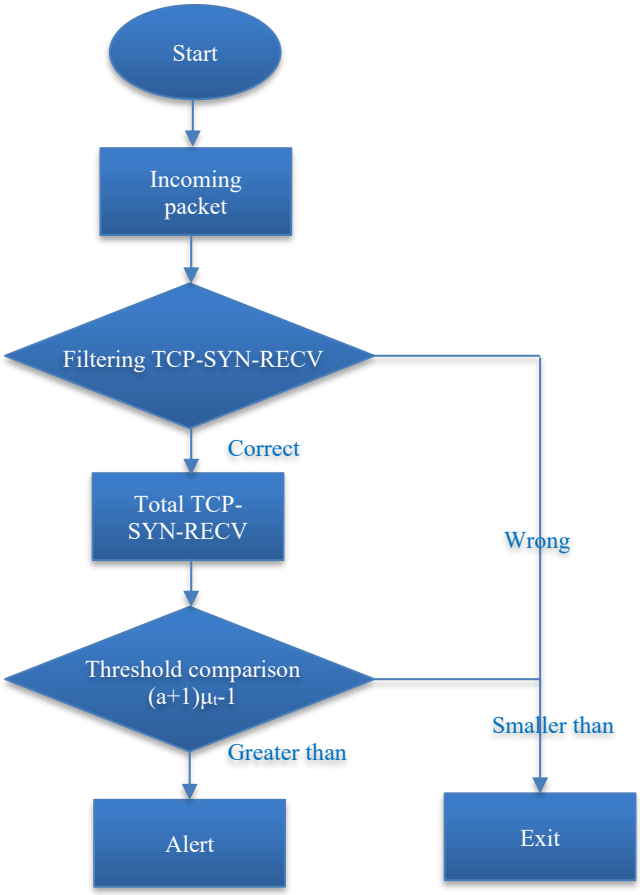

Fig. 2 Diagram of the adaptive threshold algorithm

Based on this diagram, each incoming packet is checked to determine if it is a TCP-SYN packet. If it is, the number of TCP-SYN packets is updated; then, the values of α and μ are calculated to determine the threshold. The number of TCP-SYN packets received is then compared with this threshold to determine the alert condition.

4. Results and Discussion

4.1. Normal Access Scenario

When a user accesses a service normally, the connection setup process is carried out through the three-way handshake mechanism. After the connection is successfully established and a response is received from the server, the connection remains open until the timeout period expires. Therefore, the number of TCP-SYN packets required to establish the connection again is not necessary. This characteristic is based on the HTTP 1.1 protocol standard (persistent connection), which is used by most web servers. The results collected and extracted from the TCPDUMP tool indicate that the number of TCP-SYN packets sent to the server for establishing a connection is lower than the total ratio of TCP packets.

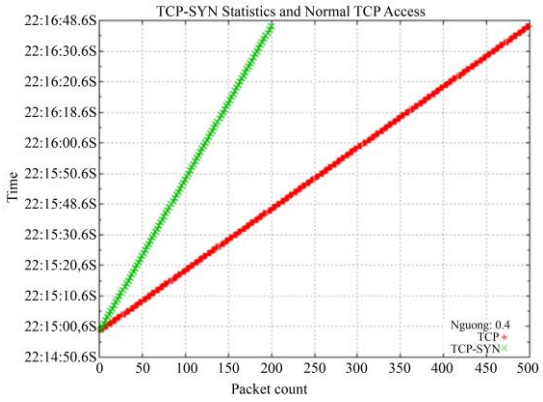

Fig. 3 Traffic connection statistics chart

The chart above shows statistical results during normal connection: The data collected after a time period (about 2 minutes) shows that the threshold value achieved is α = 0.4 (threshold defined as the percentage ratio to the average rate). This value will be applied as a safe threshold for comparison with later attack indicators. When real-time monitoring is performed using Nagios, the results indicate that this is normal access, non-threatening, and a safe alert (with code 0).

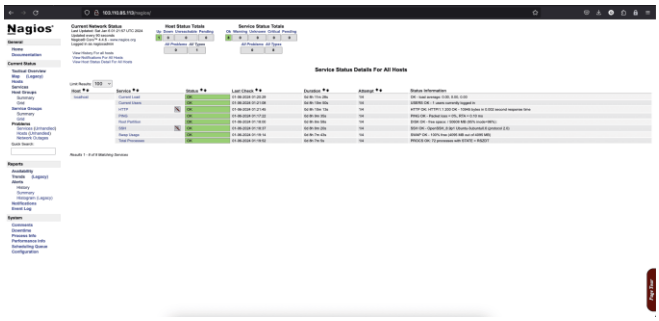

Fig. 4 Nagios alert results during normal connection

4.2. Case of Attack Occurrence

In this scenario, the number of TCP-SYN packets generated is much higher compared to normal access. The traffic may come from a single source or multiple sources, causing harm to the system. If the traffic comes from a single source, this is known as a SYN Flood Denial of Service (DoS) attack. If the TCP-SYN packets originate from multiple different sources, it is referred to as a Distributed Reflection Denial of Service (DRDoS) attack. Another scenario that can lead to a sudden increase in TCP-SYN packets, which is equally dangerous, is an application-layer attack specifically targeting the web service of a server, resulting in a reduction in server performance. This type of attack is called an HTTP-GET attack. In this attack, an attacker continuously creates a large number of successful connections and repeatedly sends requests for a page on the web server, forcing the server to maintain the connection and respond to the requested page. The number of TCP-SYN packets collected and filtered from the TCPDUMP tool indicates that the number of packets has increased drastically, signaling a potential attack Significant Increase in Traffic. This represents an abnormal increase in TCP SYN packets, indicating a potential ongoing attack. In practice, for a medium-sized system, if the number of TCP-SYN packets reaches around 500 at a given time, it is a strong indication that the system may be under attack. This abnormal traffic surge could be caused by a SYN Flood DoS attack, DRDoS, or other types of attacks that exploit the TCP handshake process, leading to the exhaustion of system resources. If this type of traffic is detected, it should trigger an immediate alert in the monitoring system (such as Nagios) to notify administrators of the suspicious activity.

In all the methods mentioned above and other similar techniques, Nagios remains capable of detecting signs of an attack. The results collected over a period of around 2 minutes show that the threshold value reached α=0.624823, which indicates that the attack was detected at very early stages. This threshold is significantly higher than the value used as a sample for normal access (where α was lower). The increase in the threshold value serves as the key indicator for detecting an attack in its early phase.

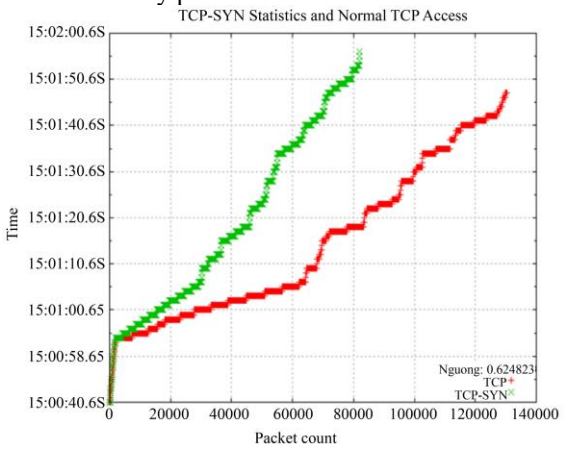

Fig. 5 Traffic diagram during an attack

The diagram above shows a significant increase in the traffic of TCP-SYN packets over time. This is an abnormal sign because, according to the principles of connection-oriented protocols, TCP-SYN packets are only used in the initial stage of the connection setup process. Once the connection is established, sending TCP-SYN messages is unnecessary. Due to the abnormal difference in the number of TCP-SYN packets, based on the threshold value, the Nagios alert feature is triggered and set with a Critical code (2). This indicates that the system has detected a possible attack or abnormal behavior, such as a SYN Flood or DRDoS.

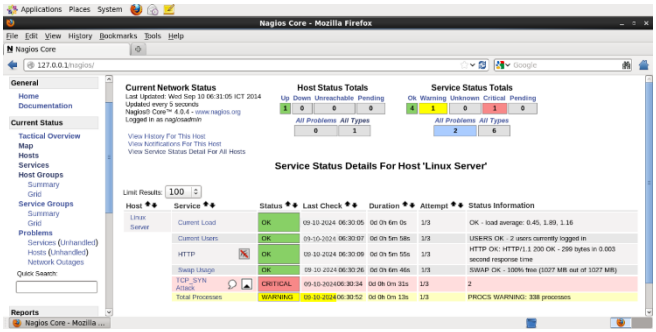

Fig. 6 Nagios alert result for attack detection

5. Conclusion

Network security issues have always been a top concern nowadays. Network systems are constantly faced with unpredictable attack threats from hackers using new methods and techniques, making it challenging to ensure the smooth operation of network systems. To effectively prevent and take appropriate measures when attacks occur, it is crucial to detect and provide timely alerts at the first sign of any attacks. The successful development of a Denial-of-Service (DoS) attack detection feature within an open-source environment has contributed significantly to the field of system security. This feature enhances the Nagios toolkit, making it more powerful and effective when integrated into the system, solving the issue raised by the thesis. This feature operates by detecting abnormal signs in the traffic volume generated during the operation of the server. A simple yet highly effective algorithm provides results right at the onset of an attack. This is a major contribution to attack detection methods because it allows abnormal behavior to be detected with high accuracy at the very first stage of the communication process. Additionally, with the existing network performance monitoring features in Nagios, administrators can obtain more complete and detailed information, making it easier to manage the system effectively.

In the future, this method could be further refined and adapted to identify other types of attacks, contributing to improving the overall security of network systems. With the combination of proactive detection and detailed real-time monitoring, systems will be better equipped to withstand attacks, reducing downtime and potential damage caused by cyber threats importance and relevance.

References

[1] Muhammad Zakarya, "DDoS Verification and Attack Packet Dropping Algorithm in Cloud Computing," *World Applied Sciences Journal*, vol. 23, no. 11, pp. 1418-1424, 2013. [Google Scholar] [Publisher Link]

[2] G.S. Navale et al., "Detecting and Analyzing DDoS Attack Using Map Reduce in Hadoop," *International Journal of Industrial Electronics and Electrical Engineering*, vol. 2, no. 2, pp. 56-58, 2014. [Google Scholar] [Publisher Link]

[3] Tongguang Zhang, "Cumulative Sum Algorithm for Detecting SYN Flooding Attacks," *Arxiv*, pp. 1-3, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[4] Haining Wang, Danlu Zhang, and Kang G. Shin, "Detecting SYN Flooding Attacks," *Proceedings Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, New York, NY, USA, pp. 1530-1539, 2002. [CrossRef] [Google Scholar] [Publisher Link]

[5] Chin-Ling Chen, and Chieh-Min Chen, "An Early Detection of Distributed Denial of Service Attack," *Advanced Computational Paradigms and Hybrid Intelligent Computing*, pp. 203-210, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[6] Thomas Chapman et al., "Design and Development of a Comprehensive Cyber Security Competition Visualization System," *Intelligent Computing*, pp. 1240-1249, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] Mohd Faris Mohd Fuzi, Nur Fatin Mohammad Ashraf, and Muhammad Nabil Fikri Jamaluddin, "Integrated Network Monitoring Using Zabbix with Push Notification via Telegram," *Journal of Computing Research and Innovation*, vol. 7, no. 1, pp. 155-163, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[8] Tom Davies et al., "A Collaborative Intrusion Detection System Using Snort IDS Nodes," *Arxiv*, pp. 1-23, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[9] Mohd Faris Mohd Fuzi et al., "Performance Analysis of Open-Source Network Monitoring Software in Wireless Network," *Journal of Computing Research and Innovation*, vol. 8, no. 2, pp. 31-44, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Subhajit Sahana et al., "Automatic Anomaly Detection by Network Traffic Analysis," *2023 3rd International Conference on Innovative Sustainable Computational Technologies*, Dehradun, India, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] John Hooft Toomey, "*An Unsupervised Based Approach to Detecting Anomalies in Hazard Monitoring Networks*," Thesis, pp. 1-38, 2024. [Google Scholar] [Publisher Link]