PCMJ

Progress in Computer and Mathematics Journal (PCMJ)
volume 3 [November, 2025]
e-ISSN: 3030-6728
Website: fskmjebat.uitm.edu.my/pcmj

NETWORK TRAFFIC MONITORING AND ATTACK DETECTION USING SNORT TOOL

MOHAMAD HAMIZAN MOHD ISHAK
College of Computing, Informatics and Mathematics UiTM Melaka, Campus Jasin, Melaka mhamizan480@gmail.com

SYAFNIDAR ABDUL HALIM*
College of Computing, Informatics and Mathematics UiTM Melaka, Campus Jasin, Melaka syafnidar@uitm.edu.my

TS. DR. MOHD AZAHARI MOHD YUSOF
College of Computing, Informatics and Mathematics UiTM Melaka, Campus Jasin, Melaka azahariyusof@uitm.edu.my

Article Info

Abstract

Snort an open-source intrusion detection and prevention system (IDS/IPS), for monitoring network traffic and detecting Distributed Denial of Service (DDoS) attacks. The research addresses the growing concern of network vulnerabilities aggravated by the emergence of sophisticated DDoS attack techniques. A key objective is to design and customized Snort rules to identify and differentiate between normal and malicious network traffic, particularly focusing on TCP SYN flood and UDP flood attacks. The project using Hping3 tool to generate various traffic scenarios, facilitating comprehensive testing in both real-world and simulated environments. Performance evaluation metrics, including detection accuracy and confusion matrix analysis, are used to validate Snort effectiveness in identifying attack patterns. Results testing that the system achieves a detection accuracy of 100%, effectively mitigating threats by triggering alerts and proactively dropping malicious traffic. Although the project successfully proves real-time traffic monitoring and DDoS detection, limitations include the focus on specific protocols and reliance on predefined rules, which may not cover more sophisticated attack methods. Future enhancements suggest integrating visualization tools like Kibana and SIEM systems such as Sguil to improve analytics and response times. This research underscores the potential of Snort as a scalable and adaptable solution for modern network security challenges.

Received: March 2025
Accepted: September 2025
Available Online: November 2025

Keywords: DDoS, Snort; IDS, IPS, Accuracy, Confusion Metric

INTRODUCTION

In recent years, cyber threats have become more complex, targeting both organizations and individuals. The Distributed Denial of Service (DDoS) are one of the most common security intrusions employed by the attackers (Sumantra & Indira Gandhi, 2020). The first recorded DDoS attack was in 1999 targeting the University of Minnesota. In recent two decades, these attacks have occurred in the more complex and large-scale form, including the famous attack of Mirai botnet in 2016, which targeted the biggest DNS providers and eliminated a few of the most popular websites (Karig & Lee, n.d.-a).

Distributed Denial of Service (DDoS) is a security threat that can be dangerous for network systems with various types of attacks that can disrupt the network. It can be started from any point and can destabilize the target server, and it is difficult for the authorized users to gain access to the network (Azahari et al., 2024). DDoS attacks can be divided into three categories: volumetric attacks, protocol - based attacks and application layer attacks (Rani et al., 2023). A DDoS attack has consequences such as financial loss, interruption of business, and damage to the organization's image.

Features such as detection systems like IDS and IPS will be implemented to detect and monitor suspicious traffic in order to mitigate threats like DDoS (Nooribakhsh & Mollamotalebi, 2020). The implementation of the Snort tool is used for the mitigation of external network packets in this project. Real time detection and monitoring will be implemented for real time detection and monitoring of suspicious activity and defending network integrity against potential threat.

LITERATURE REVIEW

DDoS known as Distributed Denial of Service attacks are a major cybersecurity threat that flood networks with excessive traffic, overwhelming servers and causing service disruptions. Attackers use botnets to send a high volume of requests, preventing legitimate users from accessing services. DDoS attacks can be categorized into three main types i.e volumetric attacks, which overload network bandwidth using large traffic floods such as UDP or ICMP floods; protocol-based attacks, which exploit network protocol vulnerabilities like TCP SYN floods; and application-layer attacks, which target web applications using techniques such as HTTP floods.

Mitigation strategies for DDoS attacks fall into proactive and reactive approaches. Proactive methods focus on continuous monitoring and anomaly detection to prevent attacks before any impact occurs, while reactive techniques detect and block attacks after malicious traffic has been identified. Packet analyzers such as Snort play a crucial role in network security by analyzing packets in real-time to identify and respond to suspicious activities.

Snort is an open-source Intrusion Detection and Prevention System (IDS/IPS) that operates in multiple modes. In sniffer mode, Snort captures and monitors network traffic. In packet logging mode, network data is stored for later analysis. In intrusion detection mode, suspicious traffic is identified, and alerts are generated. A key feature of Snort is the ability to use custom rules to detect threats. These rules include alert rules, which notify administrators of potential attacks, and drop rules, which proactively block malicious traffic such as TCP SYN floods and UDP floods.

DDoS attacks continue to pose significant challenges in cybersecurity, increasing the need for real-time traffic monitoring and detection tools. Snort provides a flexible, efficient, and effective solution for mitigating these attacks by leveraging signature-based detection and custom rule configurations.

METHODOLOGY

Methodology framework is processed to evaluate a network traffic monitoring and attack detection system using Snort. The methodology consists of five key phases i.e information gathering, design, implementation, testing, and documentation.

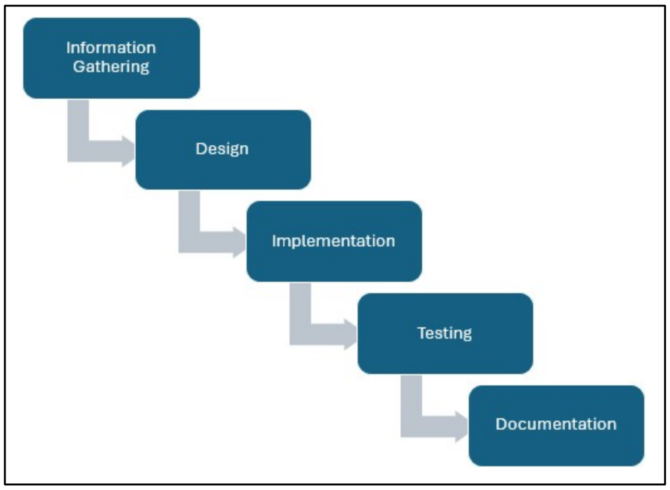


Figure 1 Methodology Framework

The Information Gathering phase involves gathering relevant information about the project being undertaken. This includes analysis process where information is gathered to know the scope and requirements. It consists of specifying the objectives, understanding the main problem, and getting detailed specifications from available resources

The Design Phase is focuses on structuring the system for real-time monitoring and detection of network anomalies and potential DDoS attacks. This phase includes the development of a logical design, which defines the overall architecture, components, and data flow of the system. Additionally, it involves the physical setup, specifying the required hardware, software, and network configurations. A detailed flowchart is also created to visually represent the detection and response process, illustrating how packets are analyzed and classified in real time.

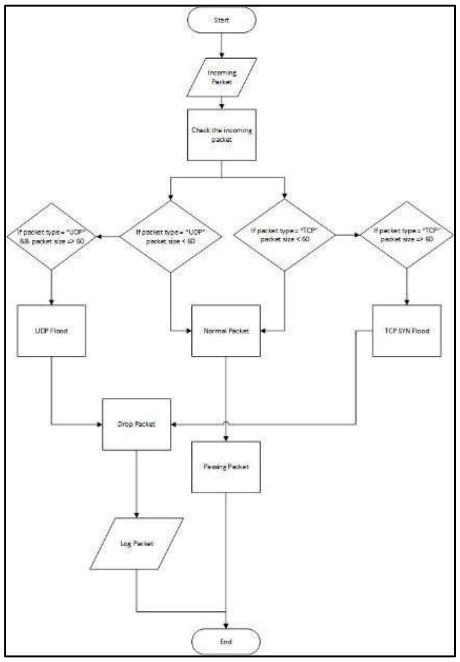

Start
Incoming Packet
Check the incoming packet
If packet type = "UDP" && packet size => 60
If packet type = "UDP" packet size < 60
If packet type = "TCP" packet size < 60
If packet type = "TCP" packet size => 60
UDP Flood
Normal Packet
TCP SYN Flood
Drop Packet
Passing Packet
Log Packet
End

Figure 2 Flowchart

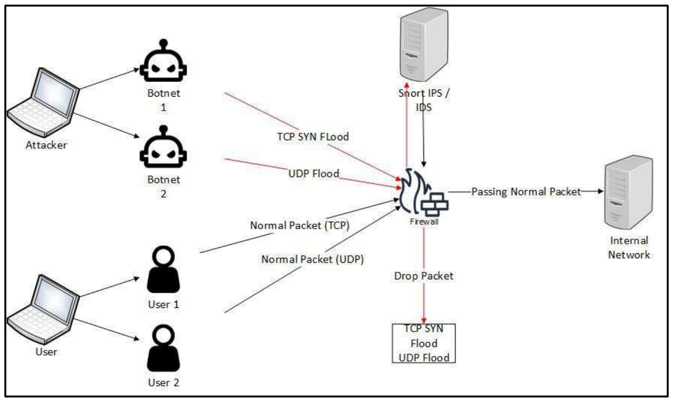


Figure 3 Logical Diagram

The implementation phase focuses on designing the system to monitor and detect DDoS attacks within network traffic. This phase involves several key activities, including configuring Snort rules to identify specific attack traffic patterns, setting up the necessary hardware and software components to ensure that the system operates effectively in a real-time environment.

The Test Phase is attacking simulation using protocol TCP and UDP with Hping3, baseline monitoring to observe normal traffic behavior. Include, execution of Snort rules to measure system performance before and after application, and performance metrics using detection accuracy, confusion matrix and log analysis to validate results.

Finally, the Documentation Phase involves recording all configurations, test results, and performance evaluations for future reference and system improvements. The methodology ensures systematic implementation, monitoring, and evaluation of Snort-based traffic monitoring. The system's efficiency in detecting and mitigating DDoS attacks is validated through real-time testing and analysis.

CONFIGURATION, SETTING UP AND TESTING THE SYSTEM

The process implementation of Snort, to monitoring and detecting network traffic anomalies will be cover the configuration, setup and testing the Snort performance evaluation in a simulated network environment.

Assigned The Unique IP Address

The firstly, setting up the network environment by configuration assigns an IP address to each host involved in project, ensuring that every device on the network has a unique identifier. This configuration is important for demonstrating that the packets being captured come from multiple sources with different IP addresses.

Setting Up the File Snort

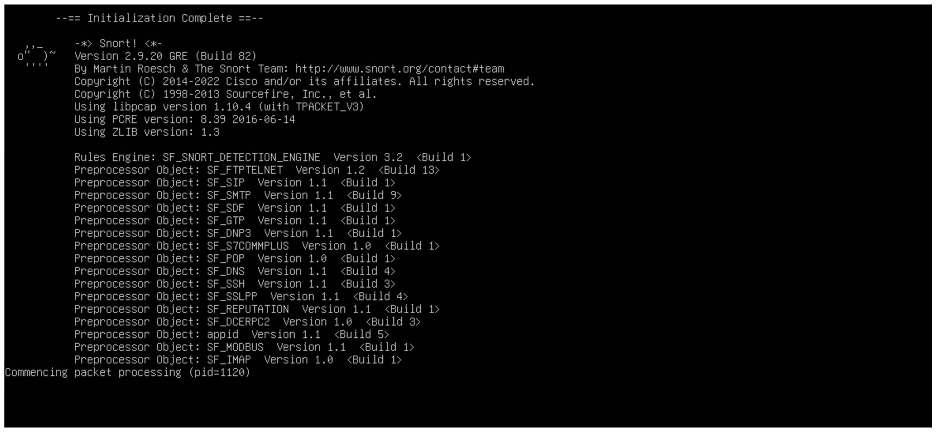
This command ***"/etc/snort/snort.conf"*** is the main configuration file which that control Snort during operation. It includes defining the network interface to monitor, define the path for store packet that capture during testing and other setting. While all packet that captured in network traffic was stored in ***'/var/log/snort/'*** for analysis.

Implementation Of Snort

The research focus on implement Snort rules to differentiate between normal and malicious traffic. In this research only focus on two rules i.e alert and drop packet. Alert rules were used to trigger and give respond about suspicious activity, while drop rules were designed to drop malicious packets before packet reach the destination. Additionally, in this research also focus on using packet threshold or packet filter which if type packet is TCP or UDP packet and size packet is equal and larger than 60 bytes/sec consider as DDoS attack.

Testing Phase

Testing involved generating TCP SYN flood and UDP flood attacks using Hping3. The project evaluates in two situations, i.e, before implementing Snort rules, which all traffic is allowed through the network and reach to the destination. While after implementing Snort rules, normal traffic was allowed passing to destination while malicious packets were successfully drop and store in log.

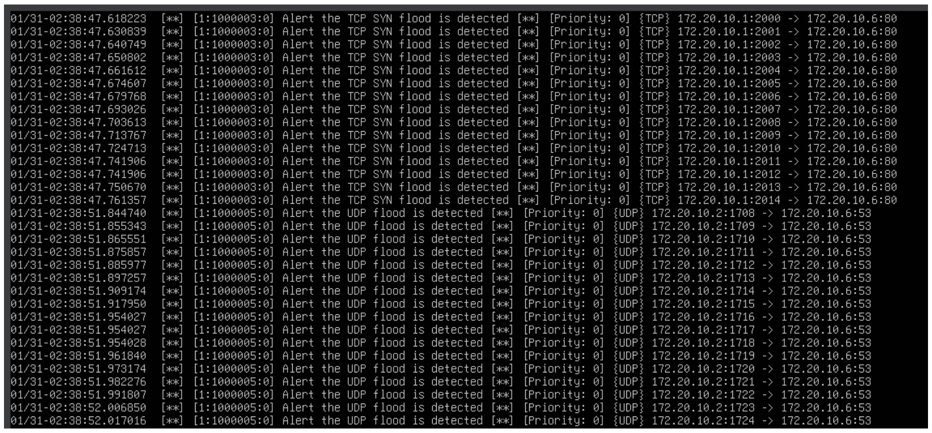


```
        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.20 GRE (Build 82)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.10.4 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.3

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.2  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_S7COMMPLUS  Version 1.0  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: appid  Version 1.1  <Build 5>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
Commencing packet processing (pid=1120)
```

Figure 4 Before Implement Snort Rules



```
01/31-02:38:47.618223  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2000 -> 172.20.10.6:80
01/31-02:38:47.630839  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2001 -> 172.20.10.6:80
01/31-02:38:47.640749  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2002 -> 172.20.10.6:80
01/31-02:38:47.650802  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2003 -> 172.20.10.6:80
01/31-02:38:47.661612  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2004 -> 172.20.10.6:80
01/31-02:38:47.674607  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2005 -> 172.20.10.6:80
01/31-02:38:47.679768  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2006 -> 172.20.10.6:80
01/31-02:38:47.693026  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2007 -> 172.20.10.6:80
01/31-02:38:47.703613  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2008 -> 172.20.10.6:80
01/31-02:38:47.713767  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2009 -> 172.20.10.6:80
01/31-02:38:47.724713  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2010 -> 172.20.10.6:80
01/31-02:38:47.741906  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2011 -> 172.20.10.6:80
01/31-02:38:47.741906  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2012 -> 172.20.10.6:80
01/31-02:38:47.750670  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2013 -> 172.20.10.6:80
01/31-02:38:47.761357  [**] [1:1000003:0] Alert the TCP SYN flood is detected [**] [Priority: 0] {TCP} 172.20.10.1:2014 -> 172.20.10.6:80
01/31-02:38:51.844740  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1708 -> 172.20.10.6:53
01/31-02:38:51.855343  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1709 -> 172.20.10.6:53
01/31-02:38:51.865551  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1710 -> 172.20.10.6:53
01/31-02:38:51.875857  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1711 -> 172.20.10.6:53
01/31-02:38:51.885977  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1712 -> 172.20.10.6:53
01/31-02:38:51.897257  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1713 -> 172.20.10.6:53
01/31-02:38:51.909174  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1714 -> 172.20.10.6:53
01/31-02:38:51.917950  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1715 -> 172.20.10.6:53
01/31-02:38:51.954027  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1716 -> 172.20.10.6:53
01/31-02:38:51.954027  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1717 -> 172.20.10.6:53
01/31-02:38:51.954028  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1718 -> 172.20.10.6:53
01/31-02:38:51.961840  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1719 -> 172.20.10.6:53
01/31-02:38:51.973174  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1720 -> 172.20.10.6:53
01/31-02:38:51.982276  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1721 -> 172.20.10.6:53
01/31-02:38:51.991807  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1722 -> 172.20.10.6:53
01/31-02:38:52.006850  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1723 -> 172.20.10.6:53
01/31-02:38:52.017016  [**] [1:1000005:0] Alert the UDP flood is detected [**] [Priority: 0] {UDP} 172.20.10.2:1724 -> 172.20.10.6:53
```

Figure 5 After Implement Snort Rules

Performance Evaluation

To calculate accuracy, classification rules must first be defined: if the packet type is TCP or UDP and the packet size is greater than or equal to 60 bytes per second, it is considered a DDoS attack. After defining the classification rules, a confusion matrix must be created before calculating accuracy. The confusion matrix consists of:

- TP (True Positive): Attack packets correctly classified as attacks = 222 (packets dropped after applying detection rules)
- TN (True Negative): Normal packets correctly classified as normal = 372 - TP = 372 - 222 = 150 (packets analyzed that were normal)
- FP (False Positive): Normal packets incorrectly classified as attacks = 0 (assuming all analyzed packets were normal)
- FN (False Negative): Attack packets incorrectly classified as normal = 0 (assuming no attacks were missed)

Calculate of Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Figure 6 Formula to Calculate Accuracy

$$Accuracy = \frac{222 + 150}{222 + 150 + 0 + 0} = \frac{372}{372} = 1$$

Result and Discussion

Based on confusion matrix, which shows an accuracy rate of 100%, which means that Snort successfully identifies and drops attack packets. Traffic analysis revealed that before implementing the Snort rule, 372 packets were analyzed and none were blocked, while after implementing the rule, 222 packets were analyzed, and 222 attack packets were blocked, showing the effectiveness of the system in detecting and preventing DDoS attacks.

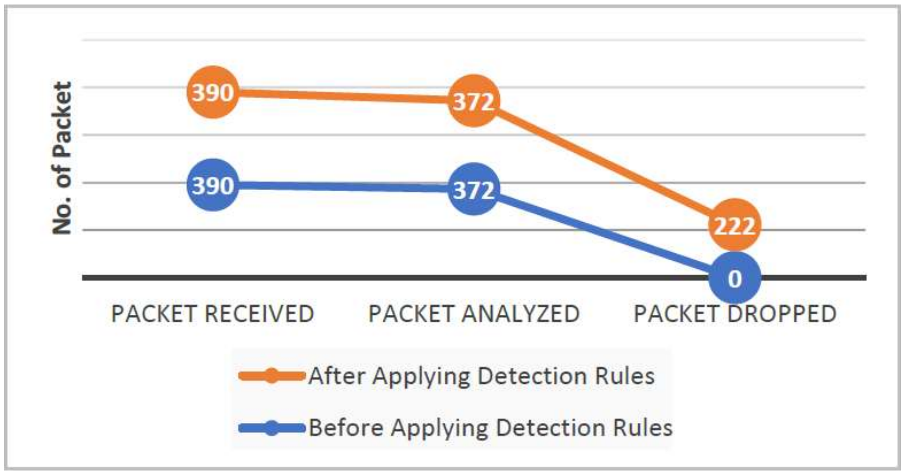


Figure 7 Graph of Accuracy

Snort proved to be an effective tool for real-time network traffic monitoring, successfully detecting and dropped packet DDoS attacks using custom rules. Performance testing confirmed that implementing Snort rules significantly improved network security by filtering out malicious traffic while allowing legitimate packets to pass through.

CONCLUSION, LIMITATION AND FUTURE GROWTH

Conclusion

The project successfully evaluates a network traffic detection and monitoring system using Snort, achieving significant results. Snort performance in detected SYN flood and UDP flood attacks by implementing custom rules, accurately identifying packets larger than 60 bytes/sec and packet send more than 60 consider as potential threats. The system provided real-time traffic monitoring, enabling quick identification of abnormal traffic patterns. Logs were systematically stored, and alerts were generated for suspicious traffic, supporting faster incident response. The system also proved to be efficient, capable of running on standard hardware, making it scalable for different environments.

Limitation

This project has several limitations that need to be considered. The detection mechanism is limited to specific protocols and packet sizes, which means it may not effectively detect attacks that deviate from these parameters. Additionally, more complex attack techniques are not suitable for the rules implemented in this project, as they rely on predefined conditions that may not cover advanced or evolving threats. Furthermore, this project is designed for a small-scale environment and has not been tested for large-scale networks, where high traffic volumes and diverse attack patterns could impact its performance.

Future growth

Future improvements for this project include integrating Snort with Kibana for enhanced traffic visualization and Sguil (Security Information and Event Management system) for centralized log management. By incorporating Kibana, network administrators can gain deeper insights into traffic patterns through interactive dashboards, making it easier to identify anomalies and potential threats. Additionally, Sguil can streamline alert management by aggregating and correlating security events in real time, improving incident response. The combination of Snort with Kibana and Sguil can significantly enhance detection accuracy, reduce false positives, and provide a more efficient and comprehensive approach to network security monitoring. This integration would strengthen the system's ability to adapt to evolving threats, making it more effective for large-scale network environments.

REFERENCES

AAMIR, M., & ZAIDI, M. A. (2013). A Survey on DDoS Attack and Defense Strategies: From Traditional Schemes to Current Techniques. *Interdisciplinary Information Sciences*, *19*(2), 173–200. https://doi.org/10.4036/iis.2013.173

Adedeji, K. B., Abu-Mahfouz, A. M., & Kurien, A. M. (2023). DDoS Attack and Detection Methods in Internet-Enabled Networks: Concept, Research Perspectives, and Challenges. In *Journal of Sensor and Actuator Networks* (Vol. 12, Issue 4). Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/jsan12040051

Azahari, M., Yusof, M., Zuraidin, N., Safar, M., Abdullah, Z., Ali, F., Ali, H., Amin, K., Sukri, M., Jofri, H., Mohamed, J., Halim Omar, A., Aryanie Bahrudin, I., Hatta, M., Ali, M., & Hani, M. (2024). DDoS Classification using Combined Techniques. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 15, Issue 1). www.ijacsa.thesai.org

Ding, D., Savi, M., Pederzolli, F., Campanella, M., & Siracusa, D. (2021). In-Network Volumetric DDoS Victim Identification Using Programmable Commodity Switches. *IEEE Transactions on Network and Service Management*, *18*(2), 1191–1202. https://doi.org/10.1109/TNSM.2021.3073597

Douligeris, C., & Mitrokotsa, A. (n.d.). *DDOS ATTACKS AND DEFENSE MECHANISMS: A CLASSIFICATION*.

Hilal, H., Ghafri, A., Zainal Abidin, Z., Kurbonov, K., & Yusof, R. (2019). Implementation of Intrusion Detection System using Snort. In *JOURNAL OF ADVANCED COMPUTING TECHNOLOGY AND APPLICATION (JACTA)* (Vol. 1, Issue 1).

Htay Yi, H., & Khaing Wai, K. (2024). ENHANCING PERFORMANCE OF TRAFFIC CLASSIFICATION WITH FEATURE SELECTION METHODS. *Indian Journal of Computer Science and Engineering*, *15*(2), 179–190. https://doi.org/10.21817/indjcse/2024/v15i2/241502031

John, E., Kalu, C., & Asuquo, P. (2022). Comparative Performance Analysis Of Cybersecurity Tools On A Wireless Network With WPA2 Encryption. In *Journal of Multidisciplinary Engineering Science and Technology (JMEST)* (Vol. 9). https://www.wireshark.org/download.html

Karig, D., & Lee, R. (n.d.-a). *Remote Denial of Service Attacks and Countermeasures*.

Kumar, S. (2007). *Smurf-based Distributed Denial of Service (DDoS) Attack Amplification in Internet*.

Kumar, V., & Prakash Sangwan, O. (2012). Signature Based Intrusion Detection System Using SNORT. In *International Journal of Computer Applications & Information Technology: Vol. I*. www.ijcait.com

Nooribakhsh, M., & Mollamotalebi, M. (2020). A review on statistical approaches for anomaly detection in DDoS attacks. In *Information Security Journal* (Vol. 29, Issue 3, pp. 118–133). Taylor and Francis Inc. https://doi.org/10.1080/19393555.2020.1717019

Nuiaa, R. R., Manickam, S., Alsaeedi, A. H., & Rahef Nuiaa, R. (2021). Distributed reflection denial of service attack: A critical review. *International Journal of Electrical and Computer Engineering (IJECE)*, *11*(6), 5327–5341. https://doi.org/10.11591/ijece.v11i6.pp%25p

Pei, J., Chen, Y., & Ji, W. (2019). A DDoS Attack Detection Method Based on Machine Learning. *Journal of Physics: Conference Series*, *1237*(3). https://doi.org/10.1088/1742-6596/1237/3/032040

Rani, S. V. J., Ioannou, I., Nagaradjane, P., Christophorou, C., Vassiliou, V., Charan, S., Prakash, S., Parekh, N., & Pitsillides, A. (2023). Detection of DDoS attacks in D2D communications using machine learning approach. *Computer Communications*, *198*, 32–51.

https://doi.org/10.1016/j.comcom.2022.11.013

Ribeiro, M. A., Pereira Fonseca, M. S., & de Santi, J. (2023). Detecting and mitigating DDoS attacks with moving target defense approach based on automated flow classification in SDN networks. *Computers and Security*, *134*. https://doi.org/10.1016/j.cose.2023.103462

Sanmorino, A. (2019). A study for DDOS attack classification method. *Journal of Physics: Conference Series*, *1175*(1). https://doi.org/10.1088/1742-6596/1175/1/012025

Sumantra, I., & Indira Gandhi, S. (2020, July 3). DDoS attack Detection and Mitigation in Software Defined Networks. *2020 International Conference on System, Computation, Automation and Networking, ICSCAN 2020*. https://doi.org/10.1109/ICSCAN49426.2020.9262408

Valdovinos, I. A., Pérez-Díaz, J. A., Choo, K. K. R., & Botero, J. F. (2021). Emerging DDoS attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions. In *Journal of Network and Computer Applications* (Vol. 187). Academic Press. https://doi.org/10.1016/j.jnca.2021.103093