Detection of Botnet in the IoT Network

Syeda Lamiya Mumtaz[1], *Hassan Jamil* Syed[2 3 *], *Ayman* Al-Ani[2], *Salmah* Fatah[2 3], *Ahmed K.* Al-Ani[4], and *Azeem* Khan[5]

[1] FAST School of Computing, National University of Computer and Emerging Sciences, Karachi 75030, Pakistan
[2] Faculty of Computing and Informatics, Universiti Malaysia Sabah, Jalan UMS, Kota Kinabalu 88400, Sabah, Malaysia
[3]Cyber Security Research Lab, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Jalan UMS, Kota Kinabalu 88400, Sabah, Malaysia
[4] School of Engineering Technology and Applied Science, Centennial College, Toronto, Ontario, Canada
[5] Faculty of Islamic Technology, University Islam Sultan Sharif Ali, Brunei Darussalam

Abstract. The ubiquity of Internet of Things (IoT) devices has prompted security concerns, particularly in the face of evolving botnet attacks. This paper investigates the impact of botnet attacks on IoT devices and proposes a network-based detection and prevention system employing signature and anomaly-based mechanisms. Notably, our methodology extends beyond traditional detection, focusing on proactively impeding bot creation. Leveraging a Linux-based distributed system, Security Information and Event Management (SIEM) tools, and custom rules, our approach encompasses distinct phases Preprocessing, Network Security Monitoring, Rule-based IDS System, and Analysis. Experimental results with diverse PCAP files demonstrate the efficacy of custom rules, significantly enhancing alert counts for various security aspects, including network trojan detection and privacy violations. The significant finding is the substantial increase in alert counts after the integration of custom rules, exemplified in the 1.1 GB PCAP file scenario. Network trojan detection surged from 585 to 988, emphasizing the heightened efficacy of rule-based measures. Privacy breaches and bad traffic alerts also experienced significant increments, showcasing the system's improved sensitivity and responsiveness. This finding reinforces the pivotal role of custom rules in fortifying IoT network security comprehensively.

1 Introduction

The ubiquity of Internet of Things (IoT) devices has attracted considerable attention, transforming into a vast network of internet-enabled embedded systems. Despite the manifold advantages, the surge in IoT usage brings forth challenges, notably security vulnerabilities and the potential for malware threats. The escalating sophistication of botnet attacks poses a severe risk to IoT networks[1].

* Corresponding author : shjamil@ums.edu.my

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

Recent instances, such as the 2021 'largest botnet'[2] and the 2020 Glupteba botnet[3], underscore the urgent need for proactive measures against these threats. This paper delves into the impact of botnet attacks on IoT devices and proposes a network-based detection and prevention system employing signature and anomaly-based mechanisms.

Motivated by the imperative to assess and mitigate the consequences of botnet attacks on IoT devices, this research addresses the critical security concern within the IoT landscape. The study aims to comprehend the extent of damage caused by botnet attacks on networks and explores effective protective measures.

The subsequent sections of this paper unfold a systematic exploration of the proposed methodology for detecting and preventing botnet attacks in IoT networks. Section 1 delves into the ubiquity of IoT devices and the ensuing security concerns, emphasizing the escalating threat posed by sophisticated botnet attacks. Section 2 provides an overview of related work, highlighting existing approaches and underscoring the need for a comprehensive, long-term solution. In Section 3, details the innovative methodology, outlining its distinct phases such as Preprocessing, Network Security Monitoring, Rule-based IDS System, and Analysis. Section 4 navigates through the experimental setup, elucidating the specifications of the hypervisor, dataset overview, and the installation and configuration processes. Results and discussions, presented in Section 5, offer a detailed analysis of network trojan detection across varying PCAP file scenarios, showcasing the substantial impact of custom rules on alert counts. Section 6 concludes the paper by summarizing the collective results, emphasizing the importance of custom rules in bolstering IoT network security, and outlining avenues for future research.

2 Related work

The integration of IoT into daily life is pervasive, as emphasized in [4], although the persistent security threats posed by IoT botnets necessitate robust countermeasures. The authors propose a PSI graph-based approach for IoT botnet detection, achieving an impressive 98.7% effectiveness through lightweight methods. This method, focused on detection, aligns with our approach, which extends beyond mere detection to proactively impede bot creation, providing a more comprehensive, long-term solution. Future enhancements, as discussed in [4], involving the incorporation of runtime data for file execution align with our goal of refining the detection approach through ongoing research.

In [5], principles for intrusion detection systems (IDS) in IoT are developed, presenting a Temporal Convolution Neural Network (TCNN) combined with SMOTE-NC. This solution demonstrates efficiency and effectiveness through experimentation, aligning with our commitment to enhancing detection methods. Future testing of IDS durability against attacks, as suggested in [5], resonates with our approach of evaluating the accuracy and robustness of our rule-based IDS system.

The criticality of IoT botnets is underscored in [1], which identifies four attack types and recommends integrating multiple detection methods for a robust and long-term solution. This recommendation aligns with our methodology, emphasizing the combination of various approaches for a more effective response to IoT botnets.

Addressing security issues associated with IoT devices, [6] conducts analyses on Android, iOS, and Linux devices. Our solution complements this work by providing a comprehensive environment for analyzing and simulating IoT botnet activities.

In [7], a malware detection model is proposed with an 89% identification rate for unknown malware, offering insights into effective detection strategies. While [8] introduces a two-level botnet detection technique using Siamese networks, [9] proposes a dynamic analysis for IoT malware detection with a CNN model in a cloud environment. These studies

highlight the diversity of approaches in the field, reinforcing the need for a combination of methods for comprehensive IoT botnet detection, as reflected in our methodology.

[10] explores botnet detection using deep learning algorithms, concluding that CNN, ANN, and RNN exhibit higher accuracy rates. The study suggests the implementation of IoT-BOT with RNN in the future. This aligns with our commitment to ongoing research and adaptation of methods for effective IoT botnet detection.

Methods for instant detection of vulnerable IoT devices in botnets are discussed in [11], employing a logistic regression model. [12] reveals the surge in botnets using DNS for communication, emphasizing the need for robust countermeasures against evolving threats. Our methodology, which includes rule-based detection and analysis, aligns with the call for robust countermeasures, contributing to the ongoing efforts to secure IoT networks.

3 Methodology

3.1 Overview

This section outlines the methodology employed to assess the impact of botnet attacks in an IoT network. Unlike previous research that primarily focused on detection, our approach aims for a more comprehensive, long-term solution. Rather than solely focusing on detection, our methodology centers on impeding bot creation. By combining various approaches, we seek to achieve a heightened efficacy in countering IoT botnets.

To simulate and analyze botnet activities, we utilize the latest botnet dataset in a Linux-based environment. Addressing the limitations of prior work, our methodology extends beyond detection to address the fundamental issue of preventing bot creation. Employing a Linux-based distribution system, we leverage Security Information and Event Management (SIEM) tools to create a robust environment, ensuring real-time visibility into data and critical operations.

3.2 Phases

This section explores the multi-faceted approach adopted in our methodology, unfolding distinct phases aimed at fortifying IoT network security. Figure 1 visually encapsulates these pivotal phases, each contributing significantly to the comprehensive protection of IoT ecosystems. From the initial preprocessing steps to the intricate rule-based intrusion detection system, and culminating in a meticulous analysis phase, the methodology navigates through a strategic sequence. Each phase plays a vital role in not only detecting intrusions but also establishing proactive measures to thwart the formation of botnets. This section meticulously expounds upon each phase, elucidating the tools, techniques, and objectives driving our innovative IoT security strategy."

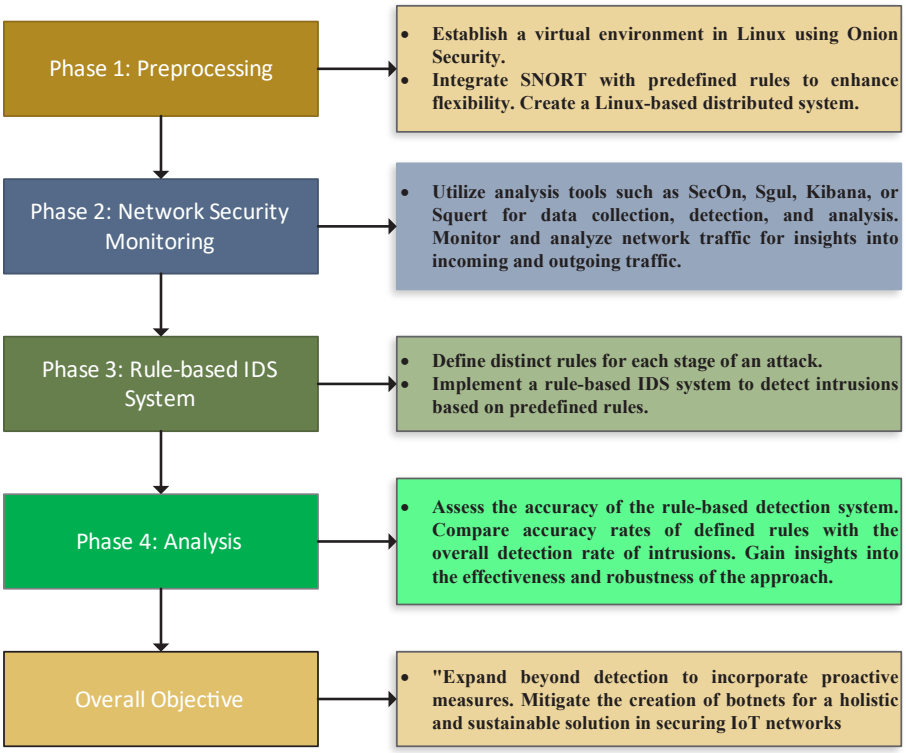

Fig. 1. Phases of the Proposed Methodology for IoT Network Security

3.2.1 Phase 1: Preprocessing

In this phase, we adopt Onion Security, a Linux-based distributed system offering a plethora of network tools. Focusing on SNORT as a primary tool, we establish a virtual environment in Linux, incorporating predefined rules to enhance flexibility and facilitate seamless integration with other tools[13].

3.2.2 Phase 2: Network Security Monitoring

For data collection, detection, and analysis, we employ analysis tools such as SecOn, Sgul, Kibana, or Squert. This phase aims to monitor and analyze network traffic, providing insights for early intervention by tracking incoming and outgoing traffic.

3.2.3 Phase 3: Rule-based IDS System

Our primary objective in this phase is to detect various attacks, irrespective of the attacker's capabilities. We achieve this by defining distinct rules for each stage of an attack. This rule-based IDS system enables the detection of intrusions based on the identification of predefined rules.

3.2.4 Phase 4: Analysis

In the final phase, we assess the accuracy of the rule-based detection system. We compare the accuracy rates of the defined rules with the overall detection rate of intrusions. This analysis provides valuable insights into the effectiveness of our approach and the robustness of the rule-based IDS system.

Expanding on previous research limitations, our methodology not only encompasses detection but also focuses on proactive measures to impede the creation of botnets, offering a more holistic and sustainable solution for securing IoT networks.

4 Experimental Setup and Solution Implementation

This section delineates the experimental setup and the subsequent implementation of the proposed solution. The outcomes of the experiment, along with an analysis of the results, are comprehensively discussed. Furthermore, a detailed overview of the Virtual Machines (VMs) involved in the experiment is provided, encompassing their specifications and role in the experimental framework. The section also includes insights into the dataset employed and the formulation of custom rules.

Security Onion[†] was selected as the framework for the experiment, serving both as the experimental environment and the platform for rule implementation. In selecting the Security Onion framework for our experimental setup, we opted for a robust and comprehensive solution that integrates a suite of cutting-edge security tools. Security Onion's open-source nature ensures transparency and allows for continuous community-driven enhancements, providing a solid foundation for our research endeavors. The framework's user-friendly interface facilitates efficient deployment and configuration of critical security components, making it accessible to researchers with varying levels of expertise. With a proven track record in the cybersecurity community, Security Onion brings reliability and scalability to our experiments, enabling us to effectively monitor, analyze, and respond to diverse network activities. The framework's adaptability and flexibility empower us to tailor configurations and rules to the specific nuances of our research objectives. Overall, Security Onion emerges as a strategic choice, aligning seamlessly with our goal of evaluating and fortifying IoT network security through advanced intrusion detection mechanisms.[14], [15] The VM was configured with specific parameters, as detailed in Table 1.

Table 1. Specifications of the Hypervisor

Specification	Details
Operating System	Windows 10
Security Onion	Version 16.04
Ram for hypervisor	8 GB
CPU cores	2

[†] https://securityonionsolutions.com/

Interfaces	2

Two interfaces were configured, one for management purposes and the other for sniffing, enabling the monitoring of inbound and outbound traffic.

4.1 Dataset Overview

In this study, we utilized a diverse dataset comprising five Packet Capture files (PCAPs) extracted from the IoT botnet dataset [14]. Each PCAP file represents a distinct capture scenario, namely CTU-IoT-Malware-Capture-44-1 (Mirai), CTU-IoT-Malware-Capture-17-1 (Kenjiro), CTU-IoT-Malware-Capture-48-1 (Mirai), CTU-IoT-Malware-Capture-7-1 (Linux_Mirai), and CTU-IoT-Malware-Capture-9-1 (Linux.Hajime). These captures encompass a wide range of network activities, including benign flows, various command and control (C&C) communications, file downloads, Distributed Denial of Service (DDoS) attacks, heartbeat signals, and port scans. The dataset provides a rich source for evaluating the proposed solution, allowing for a comprehensive analysis of network security dynamics and the effectiveness of the implemented rules. Additionally, the files vary in size, with the largest being 7.7 GB, providing a realistic representation of network traffic scenarios.

4.2 Installations and Setup

The subsequent phase involves the installation and configuration of the SNORT Network Intrusion Detection System within the Security Onion framework. Following the completion of the installation on Security Onion, the ELK stack and its associated components are deployed using Docker.

The detection engine can be configured to utilize either Suricata or SNORT. Given that SNORT is a product of Cisco 10s, we have opted for its use. The decision is motivated by the superior rules provided by Cisco 10s' threat intelligence team compared to those currently available in Suricata. Moving forward, the ELK stack is installed, where logs generated and transmitted to Logstash from Filebeat are stored in Elasticsearch and displayed on the Kibana dashboard.

Regarding updates, the Elasticsearch is upgraded to version 7.3, and SNORT is updated to version 2.9x. Custom rules for SNORT are also updated to ensure the incorporation of the latest rules available to date. Additionally, the Maximum Transmission Unit (MTU) is increased to 6400 to accommodate the larger file size of PCAPs. The PCAP file is then replayed, and the alert count is recorded.

For SNORT configuration, the sniffing interface is established, utilizing built-in rules present in its configuration file. This configuration file differs for the management interface and the SNORT interface.

4.3 Flow for Replaying Packet Capture (PCAP) Files:

After replaying each PCAP, all logs are cleared, and a new user is created. The SNORT engine is then reset, along with the user, before replaying another PCAP file. Upon system restart, the Maximum Transmission Unit (MTU) needs to be updated each time, as it reverts to 1500 after system refresh.

The following steps are iterated five times:

- Clear logs
- Restart the system

- Increase Maximum Transmission Unit (MTU) after services are started
- Replay the PCAP file

Table 2. Commands Used

Description	Command
For rule updation	so-update
To restart system	so-restart
To clear the logs	so-nsm-clear

These procedures ensure the systematic execution of the packet capture replay process, allowing for comprehensive testing and analysis.

5 Results and Discussions

This section presents the outcomes derived from the aforementioned methodology, with a focus on the comparison of collected data. The tables and graphs provided illustrate variations in alert counts when Packet Capture (PCAP) files are replayed, both before and after the integration of rules.

5.1 Network Trojan Detection in 1.1 GB PCAP File

Table 3 presents the results of replaying a 1.1 GB PCAP file before and after the addition of rules. Before the rules were implemented, a network trojan was detected 585 times. Post the rule addition, the alert count increased to 988. Privacy breaches occurred 211 times initially, escalating to 348 after the rule implementation. Alerts for bad traffic rose from 127 to 342, while miscellaneous activities increased from 104 to 198. Attempted information leaks saw an alert count increase from 60 to 67. The counts for unsuccessful attempts to gain user privileges and executable code detection remained constant.

Table 3. Network Trojan Detection in 1.1 GB PCAP File

Classification	Pre-Count	Post Count
A Network Trojan was detected	585	988
Potential Corporate Privacy Violation	211	348
Potentially Bad Traffic	127	342
Misc. activity	104	198
Attempted Information Leak	60	67
Not Suspicious Traffic	53	59
Generic Protocol Command Decode	48	53

Unsuccessful User Privilege Gain	6	6
Attempted User Privilege Gain	3	5
Executable code was detected	3	3

These findings underscore the impact of rule integration on enhancing the detection capabilities, particularly in identifying network trojans and mitigating potential security breaches. The subsequent discussions delve into the implications and significance of these results in the broader context of IoT network security.

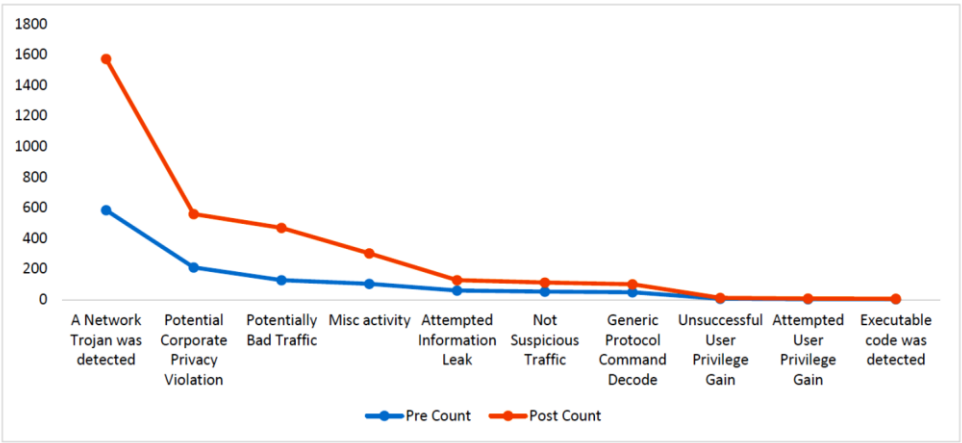

Fig. 2. Variation in Alert count of Mirai PCAP file

The analysis of results, portrayed in Figure 2, reveals the substantial impact of rule implementation on the system's detection capabilities. Notably, there is a significant increase in the alert count for network trojan detection (988), emphasizing the efficacy of rule-based measures in fortifying defenses. The rise in alerts related to potential corporate privacy violations (from 211 to 348) underscores the system's improved sensitivity to activities threatening organizational privacy. A surge in alert counts for potentially bad traffic (from 127 to 342) highlights the system's enhanced awareness and responsiveness to security threats. While miscellaneous activities and attempted information leaks show increases (104 to 198 and 60 to 67, respectively), the system remains vigilant in detecting diverse security events. Consistent alert counts for other categories indicate system stability. In summary, rule implementation significantly improves the system's adaptability and responsiveness, contributing to a more resilient IoT network security framework.

Table 4 summarizes the outcomes of replaying a 5.5 GB PCAP file, both before and after the integration of rules. Initially, network trojan detection occurred 5855 times, increasing to 988 post-rule implementation. Organization privacy breaches rose from 179 to 360. Alert counts for bad traffic elevated from 104 to 354, while miscellaneous activities increased from 89 to 198. Attempted information leaks resulted in an alert count of 79, decreasing to 67 after rule addition. Notably, the counts for unsuccessful attempts to gain user privilege and executable code detection dropped from 10 to 6. This table provides a comprehensive overview of the impact of rule integration on diverse security aspects within the 5.5 GB PCAP file scenario. Figure 2 provides a graphical view of the data discussed in Table 4.

Table 4. Results of 7.7 GB PCAP file

Classification	Pre Rule Count	Post Rule Count
A Network Trojan was detected	555	988
Potential Corporate Privacy Violation	179	360
Potentially Bad Traffic	104	354
Misc activity	89	198
Attempted Information Leak	79	67
Generic Protocol Command Decode	57	59
Not Suspicious Traffic	48	53
Unsuccessful User Privilege Gain	10	6
Attempted User Privilege Gain	6	5
Executable code was detected	3	3

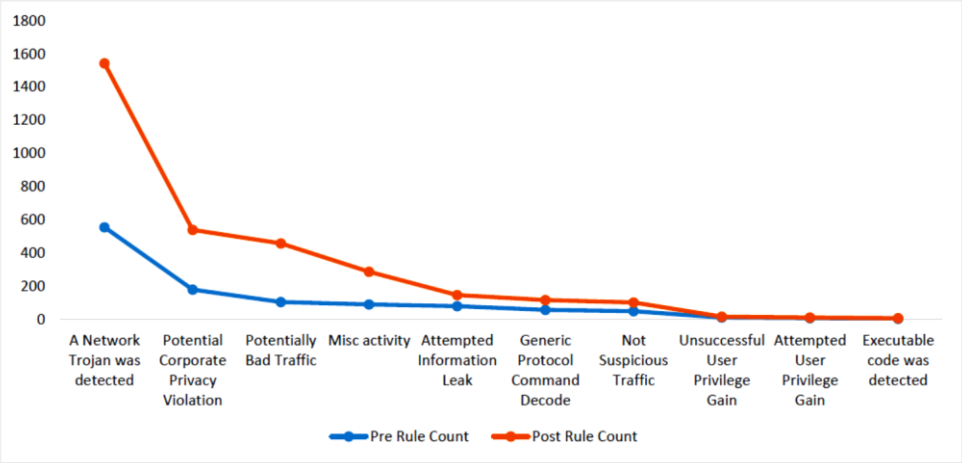

Fig. 3. Variation in Alert count of KenJiro PCAP file

Table 5 presents the outcomes of replaying a 1.7 GB PCAP file, both prior to and after the implementation of rules. Initially, network trojan detection occurred 738 times, increasing to 985 post-rule integration. Organization privacy breaches rose from 314 to 359. Alert counts for bad traffic elevated from 278 to 354, while miscellaneous activities increased from 156 to 198. Attempted information leaks resulted in an alert count of 60 before, and 67 after the addition of rules. Notably, the counts for unsuccessful attempts to gain user privilege and executable code detection remained unchanged. This table offers a detailed overview of the

impact of rule integration on various security aspects within the 1.7 GB PCAP file scenario. Figure 4 shows a graphical view of the data discussed in Table 5.

Table 5. Results of 1.7 GB PCAP file

Classification	Pre Count	Post Count
A Network Trojan was detected	738	985
Potential Corporate Privacy Violation	314	359
Potentially Bad Traffic	278	354
Misc activity	156	198
Attempted Information Leak	60	67
Generic Protocol Command Decode	57	59
Not Suspicious Traffic	53	53
Unsuccessful User Privilege Gain	6	6
Attempted User Privilege Gain	3	5
Executable code was detected	3	3

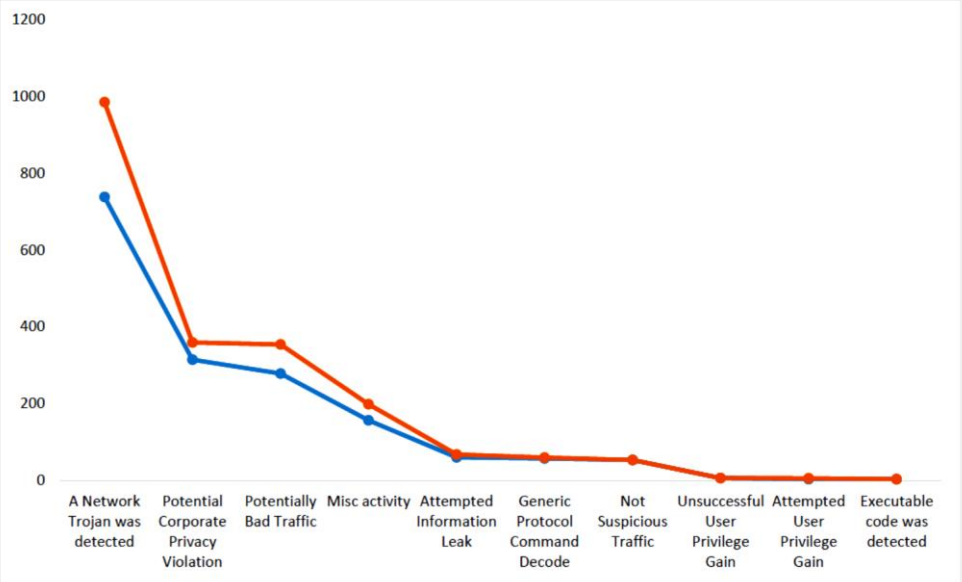

Fig. 4. Variation in alert count of Mirai (1.7 GB) PCAP

The concluding table i.e. Table 6 provides a comprehensive overview of the collective results for the files, indicating the increase in alert count following the implementation of custom

rules. Specifically, for the 1.7 GB PCAP file associated with the malware MARIA, the application of custom rules elevated the alert count from 2012 to 2050. The subsequent 5.5 GB PCAP file exhibited a rise from 2050 to 2093 after the incorporation of custom rules. Furthermore, the Linux Mirai file of 991 MB demonstrated an increase from 2030 to 2086 post-rule addition. In contrast, the Linux Hajime file of 139 MB initially recorded 1668 with built-in Snort rules, but this count surged to 2089 with the introduction of custom rules. Lastly, the Kenjiro file of 1.1 GB concluded the experiment with 1779 alerts before the application of custom rules, escalating to 2069 post-rule integration. Figure 5 Combines the of results of all the PCAPs discussed above.

Table 6. Combined Table of all the results

Malware	**Alert With BuiltinSNORT Rules**	**Alert With Custom SNORT Rules**	
Mirai	2012	2050	1.7GB
Mirai	2050	2093	5.5GB
Linux.Mirai	2030	2086	991MB
Linux.Hajime	1668	2089	139MB
Kenjiro	1779	2069	1.1GB

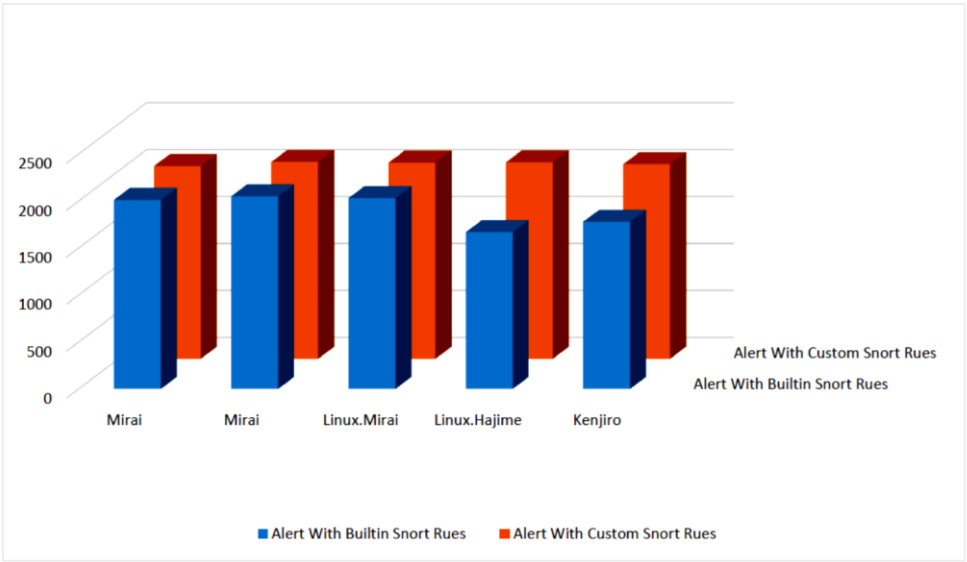

Fig. 5. Combined results of all the PCAPs

6 Conclusion & Future Work

The landscape of cyber threats continues to evolve, with various attacks targeting both organizational entities and independent users. In organizational settings, administrators employ security measures such as firewalls and Intrusion Detection Systems (IDS) to thwart

or mitigate the impact of these attacks. Security Information and Event Management (SIEM) systems, known for their versatility, find significant utility in such scenarios, notably in sectors like banking.

In the context of our research, we utilized the IoT-23 datasets and conducted experiments using SNORT community rules, augmented by additional rules tailored to IoT malwares. The implementation of custom rules yielded a notable increase in the alert count, with detailed statistics recorded before and after their integration. Specifically, in the scenario of the 1.1 GB PCAP file associated with the Mirai malware, the alert count for network trojan detection surged by approximately 69%. Similarly, the 5.5 GB PCAP file demonstrated a notable percentage increase of around 2.1% after the incorporation of custom rules. The Linux Mirai file of 991 MB exhibited a percentage rise of about 2.8%, emphasizing the adaptability of our system across different file sizes. Furthermore, the Linux Hajime file of 139 MB showcased a substantial percentage increase of approximately 25.1%. Lastly, the Kenjiro file of 1.1 GB concluded the experiment with an impressive percentage increase of around 16.4%. These findings underscore the effectiveness of our rule-based approach in significantly improving the adaptability and responsiveness of IoT network security.This enhancement in alert count proves instrumental in the timely detection of malware, enabling pre-emptive measures to be taken. Additionally, our study delves into the intricacies of selected botnet flows.

Looking forward, the future trajectory of this research involves the expansion of IoT-based rules to further strengthen the detection capabilities. The ongoing evolution of cyber threats necessitates a continuous enhancement of defense mechanisms, and the incorporation of more refined rules tailored to the nuances of IoT-based attacks is a promising avenue for future exploration.

Acknowledgement

This work has been supported by Universiti Malaysia Sabah under the grant number: SLB2241.

References

[1] S. Dange and M. Chatterjee, “IoT botnet: The largest threat to the IoT network,” in *Data Communication and Networks: Proceedings of GUCON 2019*, Springer, 2019, pp. 137–157.

[2] R. Lakshmanan, “Researchers Uncover ‘Pink’ Botnet Malware That Infected Over 1.6 Million Devices.” Accessed: Nov. 14, 2023. [Online]. Available: https://thehackernews.com/2021/11/researchers-uncover-pink-botnet-malware.html

[3] ADMIN, “Russian Botnet Attack: Over 1 Million Devices Infected,” Dec. 14, 2021. Accessed: Nov. 14, 2023. [Online]. Available: https://www.stealthlabs.com/news/over-1-million-devices-infected-in-russian-botnet-attack-warns-google/

[4] H.-T. Nguyen, Q.-D. Ngo, and V.-H. Le, “A novel graph-based approach for IoT botnet detection,” *Int. J. Inf. Secur.*, vol. 19, no. 5, pp. 567–577, Oct. 2020, doi: 10.1007/s10207-019-00475-6.

[5] A. Derhab, A. Aldweesh, A. Z. Emam, and F. A. Khan, “Intrusion detection system for internet of things based on temporal convolution neural network and efficient feature

engineering," *Wireless Communications and Mobile Computing*, vol. 2020, pp. 1–16, 2020.
[6] D. Kim, Y. Pan, and J. H. Park, "A study on the digital forensic investigation method of clever malware in IoT devices," *IEEE Access*, vol. 8, pp. 224487–224499, 2020.
[7] M. Ficco, "Detecting IoT malware by Markov chain behavioral models," presented at the 2019 IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2019, pp. 229–234.
[8] R. Vinayakumar, M. Alazab, S. Srinivasan, Q.-V. Pham, S. K. Padannayil, and K. Simran, "A visualized botnet detection system based deep learning for the internet of things networks of smart cities," *IEEE Transactions on Industry Applications*, vol. 56, no. 4, pp. 4436–4456, 2020.
[9] J. Jeon, J. H. Park, and Y.-S. Jeong, "Dynamic analysis for IoT malware detection with convolution neural network model," *IEEE Access*, vol. 8, pp. 96899–96911, 2020.
[10] S. Gaonkar, N. F. Dessai, J. Costa, A. Borkar, S. Aswale, and P. Shetgaonkar, "A survey on botnet detection techniques," presented at the 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), IEEE, 2020, pp. 1–6.
[11] A. O. Prokofiev, Y. S. Smirnova, and V. A. Surov, "A method to detect Internet of Things botnets," presented at the 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), IEEE, 2018, pp. 105–108.
[12] W. Li, J. Jin, and J.-H. Lee, "Analysis of botnet domain names for IoT cybersecurity," *IEEE Access*, vol. 7, pp. 94658–94665, 2019.
[13] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, Art. no. 3, 2019.
[14] R. Heenan and N. Moradpoor, "Introduction to security onion," 2016.
[15] A. Resmi and R. Manicka, "Intrusion detection system techniques and tools: A survey," *Scholars J. Eng. Technol.*, vol. 5, no. 3, pp. 122–130, 2017.