NETWORK PERIMETER DEFENSES USING OPEN-SOURCE SOFTWARE

Mohammed A. Al-Qassim [1], Emad H. Al-Hemiary[2]
College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
{mohammedammar[1], emad[2]}@coie-nahrain.edu.iq
Recived:22/5/2018, Accepted:30/5/2018

***Abstract*-Network security role grew exponentially in the late several years, especially with the notable amount of attacks that target all types of organizations and services. Thus, the need for more research on efficient and open source perimeter defense system and packet analysis and are rising. Intrusion detection system and firewall can afford significate role in protecting networks by detecting, reporting and blocking malicious behaviors. In this paper, an open source-based model was introduced that can provide security monitoring and logging, intrusion detection and prevention, firewall services and packet inspection and analysis. The proposed design provide high visibility and add a security layer to networks and data centers with many use cases such as: network forensics, security analysis, and production deployment. All the components and software were deployed with virtualization and container-based technology to get the benefits of these technologies like cost-effective, cloud applicable, and flexible deployment across many hardware requirements. Tests were performed to verify the performance of the implemented model. The results show that the proposed design presents detection capability of the attacks and visualization of the network traffic with security controls.**

Keywords: **Network security, Virtualization, Firewall, Intrusion Detection, Penetration Testing, Docker, Elastic Stack**.

I. INTRODUCTION

Modern-day networks face the security challenges more than any past time, while security professionals need to be right 100 percent of the time about every incident response and design decision, attackers on the other hand only need to be right once to do harm. Therefore; protecting the network require a balance between visibility and control over the network which could be achieved by implementing a layered approach of security systems and defenses. in other words, the defensive approach of the network should have a combination of proactive and reactive security measures. Control based security may refer to firewalls, Access Control Lists (ACLs), intrusion prevention, Rate limiting and Automated Response. On the other hand, visibility-based security may include full packet capture, signature-based detection, anomaly-based detection and threat intelligence.

The general network defense model consists of three main stages: prevention, detection, and reaction. The first stage is the prevention which includes security measure that prevents breaches (a security breach is any form of information security incident that results in compromising and gaining access of data, services, network applications, it happens when an unauthorized application or individual gain access to private or confidential network perimeter [1]). The prevention stage is a multi-layer approach, and this may include many phases or equipment. The Detection stage; on the other hand, is the process of monitoring and analyzing network behavior during the attack (a network attack is an attempt to get access, change, disable, or gain access to unauthorized asset [2]). The last stage is the reaction which includes the process of taking countermeasures against the ongoing attacks.

Since most datacenter network is moving into cloud and virtualization infrastructure, in many situations, the packet may not even leave the physical server, therefore; the need for a practical solution capable of providing perimeter defenses are the first thing any network security system should implement.

The perimeter defenses represent the first layer of defense in the network as part of defense in depth layered approach, which may seem old but still very practical and essential as a general model which could be used with different technologies and equipment Fig.1 shows the defense in depth layered approach. The main concept of defense in depth is to make sure that the design has many defensive layers; each layer has some security controls. This design aims to delay the attack as much as possible, and the included sensors on each layer will alert the security team to take countermeasures and kill the attack before it causes the intended damage to the network functionalities [3]. When designing perimeter security, many aspects should be in mind, such as scalability, cost, effectiveness, performance, and compatibility.

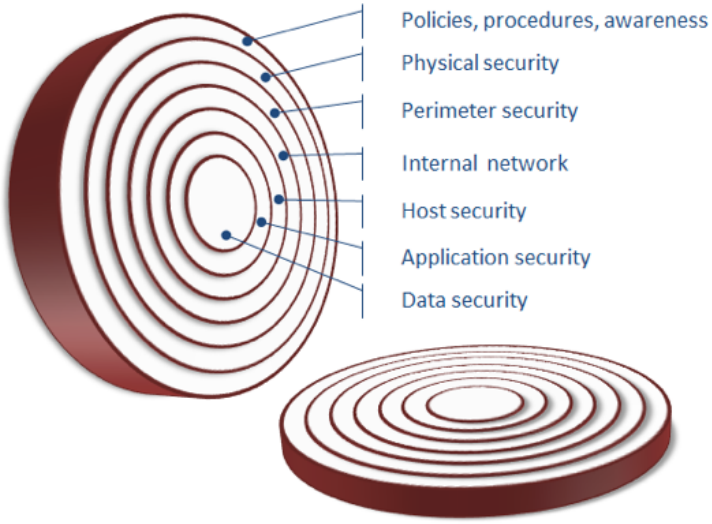

Figure 1: Defense in depth layered approach diagram

Network perimeter defense provides firewall functionality, intrusion detection and prevention systems, and monitoring and logging functionality based on open source solutions which deliver many benefits in the design requirements. The evaluation of the proposed model was made was made using vulnerability assessment and penetration testing methodologies and processes. According to the National Institute of Standards and Technology (NIST), Vulnerabilities are "software flaws or misconfigurations that cause a weakness in the security of a system" [4]. Moreover, according to MITRE foundation, it is "A mistake in software that can be directly used by a hacker to gain access to a system or network." [5]. There are many other definitions of vulnerabilities most of them have a slightly different point of view. Organizations and governments spend a tremendous amount of resources and money on researching and equipment such as firewalls, Intrusion Prevention Systems (IPS), Intrusion Detection Systems (IDS), honeypots. All for minimizing the risk of vulnerabilities in their systems.

This research suggests the use of open source solutions to implement perimeter defenses model. Virtualization and

container technology was chosen as a baseline to establish the system on. Although it could be represented in traditional hardware, the high cost and physical restrictions held significant limitations for the design, because many situations require the deployment of virtual-based security controls rather than traditional ones. This work also aims to accomplish the Open Web Application Security Project (OWASP) A10:2017 which focuses on insufficient logging and monitoring.

This research is structured as follows; the following section will include a literature review of the related work and simple explanation of their methodology. In section three, a detailed view of the proposed model and the tools used to implement it. Section four shows attack scenario using Kali Linux and results of the system response. Finally; section five contains the conclusion of the paper.

II. Literature Review

In this section, we will present a review of the related work in the field of perimeter defenses using similar tools and methodologies as proposed.

In [6], the authors include a representation of monitoring architecture that automatically runs based on security service level agreement. The architecture integrates different tools of open source and commercial products. They used OpenVAS for illustrating the detection and management of vulnerabilities as a proof of concept case study that represents their approach about improve cloud computing security, and to embrace the concept of security as a service. Their future direction is to work on the integration of other open source monitoring tools and security controls such as Snort, OSSEC into the monitoring architecture they proposed.

The research in [7] provided an overview of cyber security testing labs, using Security Onion Linux distribution in a virtual environment, they tested malware, and network traffic captures, also included honeypot and botnet example traffic. Students were embraced to use the tools provided in the proposed framework to analyze the replayed traffic captures. Also, they taught to connect tools for further investigation and analysis. The research aimed to create a framework for National Center of Academic Excellence/Cyber Defense, to show the abilities of Security Onion for researchers. It also discusses that the framework is RAM and CPU intensive. At first, they used 1GB of RAM for each Security Onion virtual machine which caused slow performance; therefore, they increased memory to 4GB for each virtual machine.

The authors in [8] proposed a lab environment for testing denial of service attacks through the use of virtualization and GNS3 network emulation. The platform was introduced for implementing ethical hacking educational courses; they utilized virtualization to increase exercises and simplify the implementation process. The lab environment used simplified graphical interfaces for practicing the denial of service attacks over simple network topology. They planned to introduce more complex environment in the future, with other types of attacks to be added, and integrate cloud technologies to get flexibility requirements.

In [9], proposed a penetration testing algorithm based on threat model, their objectives were to assemble tools and packages of penetration testing and present the model design based on extracted threat model of IT Innovative Center. Their work aimed to shorten the time consumed during the testing process so that experts could focus more on the evaluation and analysis of the given results.

The researchers in [10] presented an automation exploitation technique using single board machine and named it "Cybergrenade". The work is done with the help of virtualization and the use of nmap for scanning the network, OpenVAS for vulnerability assessment, and Metasploit for running the exploits. Attacks based on the database of Common Vulnerabilities and Exposures (CVEs), they used Kali Linux image to host the proposed system and install the required software on Cybergrenade. They used python script to automate the process of scanning and penetration testing.

The researchers in [11], studied the network traffic which is too large to process in real time for threat detection. They proposed a more efficient solution by examining the use of Artificial Neural Network (ANN) with the cybersecurity detection systems. They used Security Onion to provide labels of the cyber-attack type and the severity of the attack for the classes of the classifier. They used Snort as the main Intrusion detection system of the classifier, and Sguil console to examine and visualize threats detected by Snort. Their work aims to focus on the importance of network security monitoring and how should it evolve in the future to meet the needs. They proposed the integration of ANN with current open source solutions such as Security Onion with Snort.

This paper provides a perimeter defense model based on virtualization and containers, with use of GNS3 to create the network topology for proof of concept. Security control and visualization were achieved using Security Onion and PfSense. To test the design, attack modules were performed from Kali Linux and using offensive tools and software.

III. Proposed Perimeter Security Model

The model proposed in this paper consists of a combination of control and visibility tools, all of which are open source and free to use Fig.2 shows the proposed model topology, it has three essential network segments inside, in which could be expanded by adding more segments or zones according to the implementation requirements. A router connects the network to outside internet. Additionally, a firewall is used to provide security controls and segmentation for the topology. The segmentation aims to split the network into security zones; each zone has its private level of security control and access policy.

The first segment is for management, this is where the manager machine is located, and it has highest security level and rules to protect it, it has servers and software that connects to all other defensive security measures of the network in a distributed manner. The manager represents a network security monitoring system primary server; the system collects network alerts, logs, and packet captures; and give the user detailed information about the network security measures including incidents, analyzed captured packets, and logs. The manager gives security administrators insight into attacks attempts and breaches of the network. It is also connected to all the remaining sensors and intrusion detection engines inside the network segments.

The second segment represents a Demilitarized Zone (DMZ), where located the external services of the network. It has a low-security level and firewall rules to let the services get to the outside of the network. It hosts a web application server that represents the example of an external service that could be deployed in this segment. It also has a passive sensor that collects traffic through a physical or virtual tab or span port. The sensor is connected to the network manager, and it sends all the captured packets and alerts of security problems to be evaluated and represented to the security administrators.

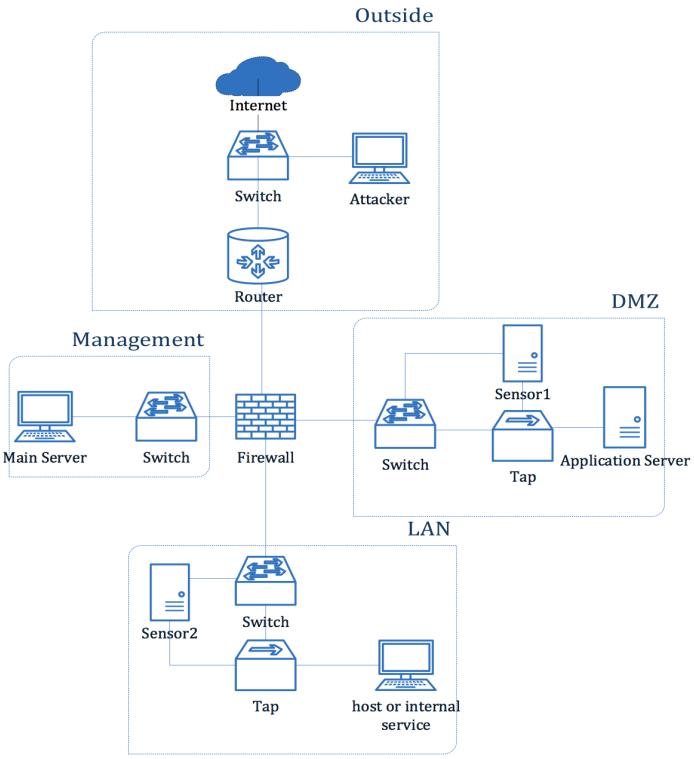

Figure 2: Proposed model topology

The third and last internal segment of the model is the inside network or LAN. It has the internal hosts and services and has a high-security level and firewall rules. It also has a sensor that connects to the Network Security Management (NSM).

All the sensors support full packet captures data, session data analysis, anomaly-based intrusion detection, Network Intrusion Detection Systems (NIDS), Network Behavior analysis, and Host Intrusion Detection System (HIDS). Each sensor has two network interfaces. The first is a management interface that has an IP address to SSH connect to the central management server and send captured data to the manager in a distributed master-sensor architecture, and the second interface is connected to tap to capture packets and analyze it in a promiscuous mode. The management system has many tools for visualization, security monitoring, and analysis that used to analyze the captured traffic and fired alerts through the network.

On each place that had a sensor installed, either a SPAN port or a network tap needs to be implemented to get the traffic to the sensor device. All the model servers, offensive and defensive hosts, and sensors could be installed using bare metal hardware, or on virtual machines using virtualization hypervisor of either type1 or type2; and connected using real network devices or network emulator. On this research, the virtualization deployment and Docker containers were chosen as an efficient and cost-effective solution for testing implementation, evaluation, and even running real-time systems.

The tools that represent visibility are included in Security Onion and installed inside Docker cluster; Security Onion is a Linux distribution created to perform NSM tasks, it provides multiple IDSs including HIDS and NIDS, network security monitoring, log management and analysis tools. To provide control and segmentation of the traffic flow in the network design, PfSense was chosen to represent Unified Threat Management (UTM) to the network, a free open source firewall based on FreeBSD. PfSense could perform multiple firewall and security functionality at the same time. In the implemented scenario, it is mainly a packet filtering firewall with zone-based functionality for fragmentation of the network segments.

The offensive security measures consist of some tools and software that could be used to test the network defense against various attack scenarios; attacker machine hosts a large number of software tools and frameworks that can be used to perform testing and attacking scenarios in the network.

In this paper, the proposed design suggests using PfSense as UTM solution with firewall functionalities which represent security control in the network. Security Onion was utilized as IDS with distributed configuration, this design choice was made to assure that each segment of the network could have its specific sensor with centralized management placed in a more secure zone. Each sensor was built using ubuntu server with minimalistic hardware requirements, docker cluster was used to implement Elastic Stack on the security onion sever and the distributed sensors across the network.

IV. Test Scenario and Results

This section contains a series of test mechanisms and defensive measures against various attack scenarios. The vast number of tools and security controls installed in the testing environment made that possible. Comparison between the response of two known intrusion detection system was made against Metasploit exploit scenario; the evaluation was using elastic stack installed on top of docker. Then a penetration testing scenario with vulnerability assessment against target web application server was made. The first test represents a comparison between two network Intrusion Detection Systems; the goal is to find which IDS engine is better to deal with ruleset and alert generation, the test was made between Snort and Suricata. Both are designed to use the Emerging Threats (ET) and Talos rulesets. The implemented environment NSM system uses both ET and Talos rulesets; a comparison was made to determine the main IDS engine that will be installed in the environment deployment.

The scenario consists of attacking machine represented by Kali Linux running Metasploit attacks using Armitage. The target is an Ubuntu-based vulnerable web application hosts represented by metasploiable2. The server will be monitored by a IDS system running Security Onion. Two attacks will be made, the first against Security Onion running Snort as the main IDS engine, and the second against Security Onion running Suricata as the main IDS engine. The NSM system will be using Elastic Stack for evaluation, which will include Elastic Search, Logstash, and Kibana. Each one of them installed on separated Docker container. The NSM system will be running in standalone mode due to hardware limitation because the minimum requirements of RAM to operate Elastic Stack is 8GB, which is relatively high considering that the total RAM of the environment is 16GB Fig.3 shows the running test attack route. During this experiment, firewall and security controls were running in routing mode only, to ensure that the attacks reach the destination and the IDS could fire alerts regarding the exploits.

The evaluation of the test was made using elastic stack analytical tools provided by Kibana, which has rich data analytical features and visualization. The number of fired rules is less in Suricata than Snort as shown in Fig.4 As a conclusion, snort is better with Talos and ET ruleset. The evaluation was made easy because of the use of Kibana which provide many visualization and data analysis solutions. Open source IDS/IPS systems such as Snort and Suricata both have proper detection and performance. However, there are some differences in the detection engine between them. For example, Suricata is a newer project that could get better in the future due to features such as multithreading, and file identification and extraction. However, it lacks documentation and needs time to get mature. On the other hand, Snort is very well documented with affluent community and support; it is also better handling the rulesets proposed during the analysis. Therefore; snort was chosen as the primary IDS engine for the proposed implementation. It will be implemented with both elastic search and docker and with production master-slave deployment. Also, results show that snort utilizes less CPU load during the tests which made it a better choice in limited resources environment such as our proposed one.

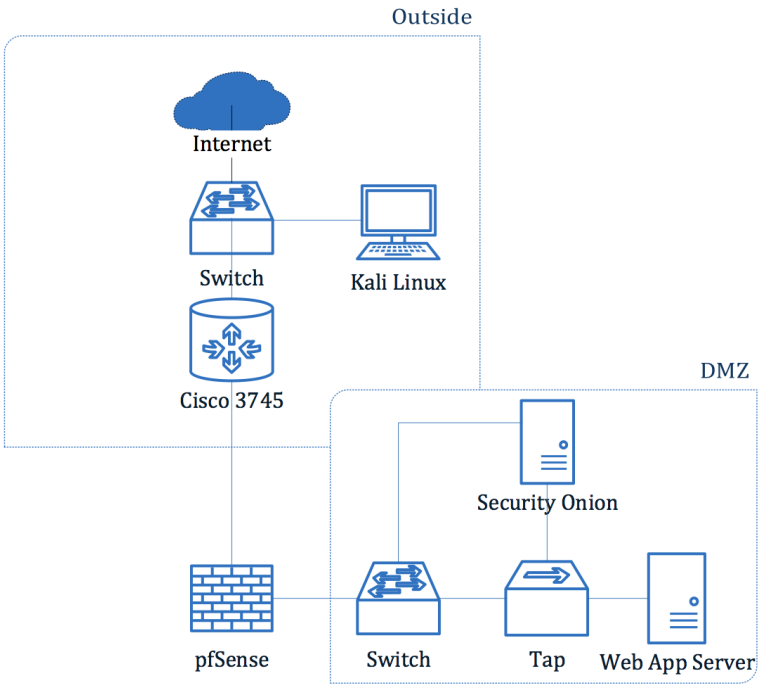

Figure 3: Attack route from kali linux to target machine

Fig.5 and Fig.6 shows detailed logs count using Snort, and Suricata. The logs were gathered and showed using Elastic Stack installed on Docker containers. Kibana was used to view the logs and alerts for data visualization and monitoring.

The next test shows vulnerability scan results of scanner against target machines; it will also show the scanner result and performance with different configuration and cases. OpenVAS was installed on Kali Linux and used for all the testing scenarios. Both credentialed, and uncredentialled scans were made against the web application server, the uncredentialled scan means that the scanner knows the login credentials using ssh, this credential was used to get access to the target.

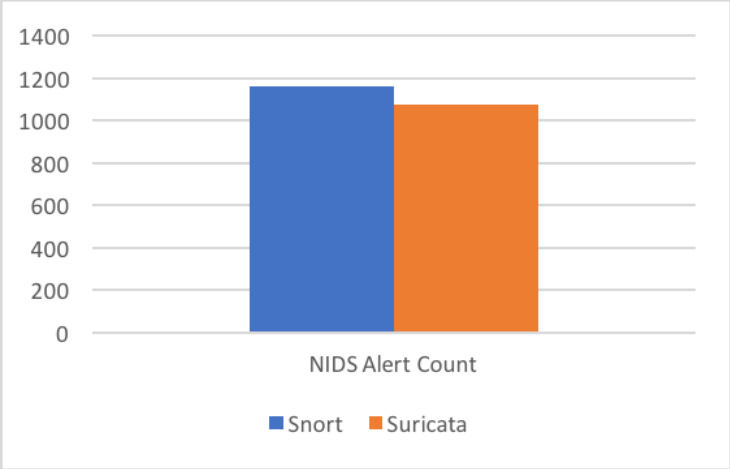

Figure 4: Number of fired alerts on snort and suricata

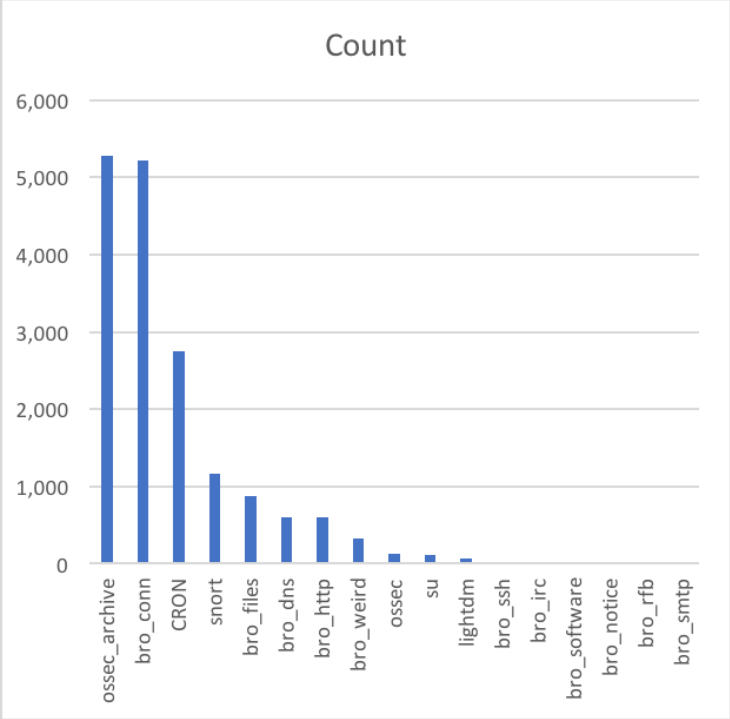

Figure 5: Snort all sensor logs and alerts using kibana

Where the uncredentialled scan means that the scanner does not know the target credentials. Each scan generated a separate report outlining the vulnerabilities and their description, also the level of severity of the vulnerability (Hight, Medium, Low) and how to mitigate if it is possible Fig.7 shows a comparison between credentialed and uncredentialled scan and the number of vulnerabilities each scan reveals.

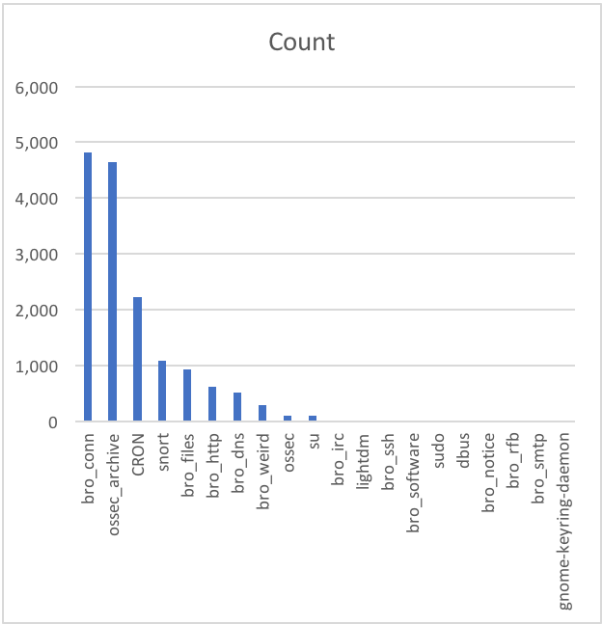

Figure 6: Suricata all sensor logs and alerts using kibana

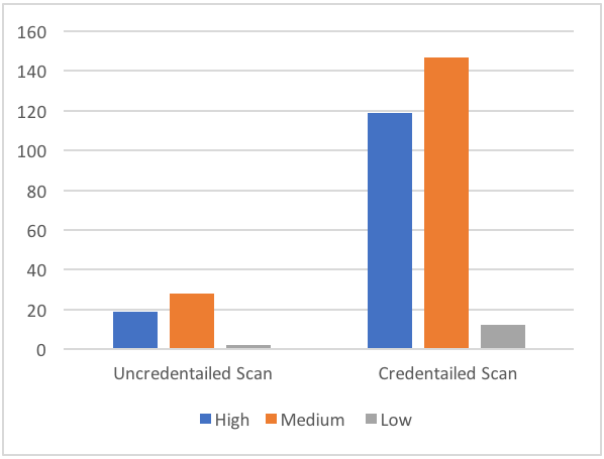

Figure 7: OpenVAS scan results against web application server

V. Conclusion and Future Work

This paper suggests and implements a perimeter defense model suited for multiple use cases such as academic research, security forensics, or even deployed in real-time organizations. Since the model focuses on visualization and logs analysis, a comparison was made between two IDS software to determine which one to use. Snort showed to be more reliable for implementation. Then a scanning was performed against different attack scenarios to show the benefits of using the proposed model defensive capabilities. The results showed an increase on security controls in the network and provided

monitoring and visualization of the network traffic compared to using traditional security or no perimeter security. The visualization provided by the software made network monitoring effective for large logs and alerts analysis to fit for use cases such as the Internet of Things and Big Data. Future work may include scaled implementation of the model design and validating performance against more attack scenarios for studying the effectiveness of the proposed model against attacks and vulnerabilities. Also, the integration of artificial intelligence and machine learning could add to the perimeter security detection capabilities.

REFERENCES

[1] M. Krausz , Managing information security breaches studies from real life. It governance Publishing , 2015.
[2] E. O. Yuri Diogenes, Cybersecurity: Attack and Defense Strategies. 2018.
[3] C. Sanders, J. Smith, and R. Bejtlich, The Practice of Network Security Monitoring, vol. 2014, no. 10. 2014.
[4] D. G. Rosado , Security engineering for cloud computing approaches and tools. IGI Global 701 E. Chocolate Avenue, Hershey, Pennsylvania, 17033, US , 2013.
[5] S. Shah and B. M. Mehtre, "An overview of vulnerability assessment and penetration testing techniques," J. Comput. Virol. Hacking Tech., vol. 11, no. 1, pp. 27 _ 49, 2015.
[6] V. Casola, A. De Benedictis, and M. Rak, "Security monitoring in the cloud: An SLA-based approach," Proc. 10th Int. Conf. Availability, Reliab. Secur. ARES 2015, pp. 749_ 755, 2015.
[7] R. Gonzales, A. Watkins, and C. Simpson, "Using Security Onion for Hands-On Cybersecurity Labs," 2015.
[8] S. Al Kaabi, N. Al Kindi, S. Al Fazari, and Z. Trabelsi, "Virtualization based ethical educational platform for hands on lab activities on DoS attacks," IEEE Glob. Eng. Educ. Conf. EDUCON, vol. 1013Apri, no. April, pp. 273_ 280, 2016.
[9] N. A. Almubairik and G. Wills, "Automated penetration testing based on a threat model," 2016 11th Int. Conf. Internet Technol. Secur. Trans. ICITST 2016, pp. 413_ 414, 2017.
[10] A. Akkiraju, D. Gabay, H. B. Yesilyurt, H. Aksu, and S. Uluagac, "Cybergrenade: Automated Exploitation of Local Network Machines via Single Board Computers," Proc. - 14th IEEE Int. Conf. Mob. Ad Hoc Sens. Syst. MASS 2017, pp. 580_ 584, 2017.
[11] K. L. Moore, T. J. Bihl, K. W. Bauer, and T. E. Dube, "Feature extraction and feature selection for classifying cyber traffic threats," J. Def. Model. Simul., vol. 14, no. 3, pp. 217_ 231, 2017.